

# EXHIBIT A

REDACTED VERSION OF DOCUMENT(S)  
SOUGHT TO BE SEALED

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**IN THE UNITED STATES DISTRICT COURT  
FOR THE NORTHERN DISTRICT OF CALIFORNIA  
SAN JOSE DIVISION**

NETFUEL, INC.,

*Plaintiff,*

v.

CISCO SYSTEMS, INC.

*Defendant.*

Civil Action No. 5:18-cv-2352-EJD

**JURY TRIAL DEMANDED**

**OPENING EXPERT REPORT OF DR. KEVIN C. ALMEROOTH ON INVALIDITY OF  
U.S. PATENT NOS. 7,747,730 AND 9,663,659**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**CONTENTS**

|              |  |           |
|--------------|--|-----------|
| <b>I.</b>    | <b>BACKGROUND AND QUALIFICATIONS .....</b>   | <b>1</b>  |
| A.           | Background.....  | 1         |
| B.           | Previous Expert Witness Experience .....   | 6         |
| C.           | Compensation .....   | 6         |
| <b>II.</b>   | <b>MATERIALS CONSIDERED.....</b>   | <b>6</b>  |
| <b>III.</b>  | <b>SUMMARY OF OPINIONS.....</b>  | <b>7</b>  |
| <b>IV.</b>   | <b>LEGAL STANDARDS .....</b>   | <b>7</b>  |
| A.           | Anticipation and Obviousness .....   | 7         |
| B.           | The “Written Description” Requirement .....  | 8         |
| C.           | The “Enablement” Requirement.....  | 9         |
| D.           | Definiteness.....  | 10        |
| E.           | Inventorship .....   | 10        |
| F.           | Subject Matter Eligibility.....  | 10        |
| G.           | Level of Ordinary Skill in the Art.....  | 10        |
| <b>V.</b>    | <b>BACKGROUND OF THE TECHNOLOGY AND ASSERTED PATENTS.....</b>  | <b>11</b> |
| A.           | The Asserted Patents Concern Automated Management of Large Enterprise Networks.....  | 11        |
| B.           | The Asserted Patents Describe a Hierarchical Network Management System.....  | 12        |
| C.           | Priority Dates .....   | 16        |
| <b>VI.</b>   | <b>THE ASSERTED CLAIMS .....</b>   | <b>17</b> |
| <b>VII.</b>  | <b>CLAIM CONSTRUCTION .....</b>  | <b>17</b> |
| <b>VIII.</b> | <b>ANTICIPATORY PRIOR ART FOR THE '659 PATENT .....</b>  | <b>19</b> |
| A.           | Anticipation by and/or Obviousness in View of United States Patent No. 6,460,070, filed 6/3/1998 and issued to Turek, et al. on 10/2/2002..... | 19        |
| 1.           | Claim 1 .....  | 19        |
| 2.           | Claim 2.....   | 27        |
| 3.           | Claim 3.....   | 28        |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |    |
|-----|---|----|
| 4.  | Claim 6.....  | 29 |
| 5.  | Claim 7.....  | 29 |
| 6.  | Claim 8.....  | 31 |
| 7.  | Claim 9.....  | 31 |
| 8.  | Claim 13.....   | 31 |
| 9.  | Claim 14.....   | 33 |
| 10. | Claim 15.....   | 33 |
| 11. | Claim 18.....   | 34 |
| B.  | Anticipation by and/or Obviousness in View of Great Britain Patent No. 2<br>363 284, issued to Goldman, et al. on 12/12/2001.....                     | 34 |
| 1.  | Claim 1.....  | 34 |
| 2.  | Claim 2.....  | 38 |
| 3.  | Claim 3.....  | 39 |
| 4.  | Claim 6.....  | 40 |
| 5.  | Claim 7.....  | 40 |
| 6.  | Claim 8.....  | 42 |
| 7.  | Claim 9.....  | 42 |
| 8.  | Claim 13.....   | 43 |
| 9.  | Claim 14.....   | 45 |
| 10. | Claim 15.....   | 45 |
| 11. | Claim 18.....   | 46 |
| C.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,671,724, filed 3/21/2000 and issued to Pandya, et al. on 12/30/2003. .... | 46 |
| 1.  | Claim 1.....  | 46 |
| 2.  | Claim 2.....  | 51 |
| 3.  | Claim 3.....  | 52 |
| 4.  | Claim 6.....  | 52 |
| 5.  | Claim 7.....  | 53 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |    |
|-----|--|----|
| 6.  | Claim 8.....   | 55 |
| 7.  | Claim 9.....   | 55 |
| 8.  | Claim 13.....  | 55 |
| 9.  | Claim 14.....  | 58 |
| 10. | Claim 15.....  | 58 |
| 11. | Claim 18.....  | 58 |
| D.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,012,152, issued to Douik, et al. on 1/4/2000. ....                         | 59 |
| 1.  | Claim 1.....   | 59 |
| 2.  | Claim 2.....   | 63 |
| 3.  | Claim 3.....   | 63 |
| 4.  | Claim 6.....   | 64 |
| 5.  | Claim 7.....   | 65 |
| 6.  | Claim 8.....   | 66 |
| 7.  | Claim 9.....   | 66 |
| 8.  | Claim 13.....  | 67 |
| 9.  | Claim 14.....  | 69 |
| 10. | Claim 15.....  | 70 |
| 11. | Claim 18.....  | 70 |
| E.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,584,502, filed 6/29/1999 and issued to Natarajan, et al. on 6/24/2003..... | 70 |
| 1.  | Claim 1.....   | 70 |
| 2.  | Claim 2.....   | 75 |
| 3.  | Claim 3.....   | 75 |
| 4.  | Claim 6.....   | 76 |
| 5.  | Claim 7.....   | 76 |
| 6.  | Claim 8.....   | 78 |
| 7.  | Claim 9.....   | 78 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 8.  | Claim 13.....   | 79  |
| 9.  | Claim 14.....   | 81  |
| 10. | Claim 15.....   | 82  |
| 11. | Claim 18.....   | 82  |
| F.  | Anticipation by and/or Obviousness in View of United States Patent No. 5,655,081, issued to Bonnell, et al. on 8/5/1997.....    | 82  |
| 1.  | Claim 1.....  | 82  |
| 2.  | Claim 2.....  | 86  |
| 3.  | Claim 3.....  | 87  |
| 4.  | Claim 6.....  | 87  |
| 5.  | Claim 7.....  | 88  |
| 6.  | Claim 8.....  | 89  |
| 7.  | Claim 9.....  | 90  |
| 8.  | Claim 13.....   | 90  |
| 9.  | Claim 14.....   | 92  |
| 10. | Claim 15.....   | 93  |
| 11. | Claim 18.....   | 93  |
| G.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,122,664, issued to Boukobza, et al. on 9/19/2000. .... | 93  |
| 1.  | Claim 1.....  | 93  |
| 2.  | Claim 2.....  | 97  |
| 3.  | Claim 3.....  | 98  |
| 4.  | Claim 6.....  | 99  |
| 5.  | Claim 7.....  | 99  |
| 6.  | Claim 8.....  | 101 |
| 7.  | Claim 9.....  | 101 |
| 8.  | Claim 13.....   | 101 |
| 9.  | Claim 14.....   | 104 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 10. | Claim 15.....  | 104 |
| 11. | Claim 18.....  | 104 |
| H.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,665,262, filed 2/16/1999 and issued to Lindskog, et al. on 12/16/2003.....  | 104 |
| 1.  | Claim 1.....   | 105 |
| 2.  | Claim 2.....   | 109 |
| 3.  | Claim 3.....   | 110 |
| 4.  | Claim 6.....   | 110 |
| 5.  | Claim 7.....   | 111 |
| 6.  | Claim 8.....   | 113 |
| 7.  | Claim 9.....   | 113 |
| 8.  | Claim 13.....  | 114 |
| 9.  | Claim 14.....  | 116 |
| 10. | Claim 15.....  | 117 |
| 11. | Claim 18.....  | 117 |
| I.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,675,128, filed 9/30/1999 and issued to Hellerstein, et al. on 1/6/2004..... | 117 |
| 1.  | Claim 1.....   | 117 |
| 2.  | Claim 2.....   | 122 |
| 3.  | Claim 3.....   | 122 |
| 4.  | Claim 6.....   | 123 |
| 5.  | Claim 7.....   | 123 |
| 6.  | Claim 8.....   | 125 |
| 7.  | Claim 9.....   | 125 |
| 8.  | Claim 13.....  | 126 |
| 9.  | Claim 14.....  | 128 |
| 10. | Claim 15.....  | 128 |
| 11. | Claim 18.....  | 129 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| J.  | Anticipation by and/or Obviousness in View of <i>Distributed Management with Mobile Components</i> , by Kasteleijn, et al., published in the May 1999 Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management. ....   | 129 |
| 1.  | Claim 1.....  | 129 |
| 2.  | Claim 2.....  | 133 |
| 3.  | Claim 3.....  | 133 |
| 4.  | Claim 6.....  | 133 |
| 5.  | Claim 7.....  | 134 |
| 6.  | Claim 8.....  | 135 |
| 7.  | Claim 9.....  | 136 |
| 8.  | Claim 13.....   | 136 |
| 9.  | Claim 14.....   | 139 |
| 10. | Claim 15.....   | 140 |
| 11. | Claim 18.....   | 140 |
| K.  | Anticipation by and/or Obviousness in View of <i>Proactive Management of Computer Networks using Artificial Intelligence Agents and Techniques</i> , by da Rocha, et al., published in the 1997 issue of Integrated Network Management V (Springer). .... | 140 |
| 1.  | Claim 1.....  | 141 |
| 2.  | Claim 2.....  | 146 |
| 3.  | Claim 3.....  | 146 |
| 4.  | Claim 6.....  | 147 |
| 5.  | Claim 7.....  | 148 |
| 6.  | Claim 8.....  | 150 |
| 7.  | Claim 9.....  | 150 |
| 8.  | Claim 13.....   | 150 |
| 9.  | Claim 14.....   | 153 |
| 10. | Claim 15.....   | 153 |
| 11. | Claim 18.....   | 153 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| L.  | Anticipation by and/or Obviousness in View of <i>Proactive Management of Software Aging</i> , by Castelli, et al., published in March 2001 by IBM. ....  | 154 |
| 1.  | Claim 1 .....  | 154 |
| 2.  | Claim 2 .....  | 160 |
| 3.  | Claim 3 .....  | 161 |
| 4.  | Claim 6 .....  | 163 |
| 5.  | Claim 7 .....  | 164 |
| 6.  | Claim 8 .....  | 166 |
| 7.  | Claim 9 .....  | 166 |
| 8.  | Claim 13 .....   | 166 |
| 9.  | Claim 14 .....   | 169 |
| 10. | Claim 15 .....   | 169 |
| 11. | Claim 18 .....   | 169 |
| M.  | Anticipation by and/or Obviousness in View of <i>Network Security Management with Intelligent Agents - A First Step with SLD ("Conti")</i> , published in the 1998 International Workshop on Intelligent Agents for Telecommunication Applications. .... | 170 |
| 1.  | Claim 1 .....  | 170 |
| 2.  | Claim 2 .....  | 178 |
| 3.  | Claim 3 .....  | 179 |
| 4.  | Claim 6 .....  | 180 |
| 5.  | Claim 7 .....  | 181 |
| 6.  | Claim 8 .....  | 183 |
| 7.  | Claim 9 .....  | 183 |
| 8.  | Claim 13 .....   | 183 |
| 9.  | Claim 14 .....   | 186 |
| 10. | Claim 15 .....   | 187 |
| 11. | Claim 18 .....   | 187 |
| N.  | Anticipation by and/or Obviousness in View of <i>Network Security Management with Intelligent Agents ("NSMIA")</i> , NOMS 2000. 2000   |     |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
|     | IEEE/IFIP Network Operations and Management Symposium 'The Networked Planet: Management Beyond 2000. ....  | 187 |
| 1.  | Claim 1.....   | 188 |
| 2.  | Claim 2.....   | 199 |
| 3.  | Claim 3.....   | 199 |
| 4.  | Claim 6.....   | 200 |
| 5.  | Claim 7.....   | 201 |
| 6.  | Claim 8.....   | 204 |
| 7.  | Claim 9.....   | 204 |
| 8.  | Claim 13.....  | 204 |
| 9.  | Claim 14.....  | 209 |
| 10. | Claim 15.....  | 209 |
| 11. | Claim 18.....  | 209 |
| O.  | Anticipation by and/or Obviousness in View of <i>INCA: An Agent-based Network Control Architecture</i> , published in the 1998 International Workshop on Intelligent Agents for Telecommunication Applications. .... | 210 |
| 1.  | Claim 1.....   | 210 |
| 2.  | Claim 2.....   | 215 |
| 3.  | Claim 3.....   | 215 |
| 4.  | Claim 6.....   | 216 |
| 5.  | Claim 7.....   | 217 |
| 6.  | Claim 8.....   | 218 |
| 7.  | Claim 9.....   | 219 |
| 8.  | Claim 13.....  | 219 |
| 9.  | Claim 14.....  | 222 |
| 10. | Claim 15.....  | 222 |
| 11. | Claim 18.....  | 222 |
| P.  | Anticipation by and/or Obviousness in View of <i>Distributed Network Security Management Using Intelligent Agents</i> , by Boudaoud, et al.,   |     |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
|     | published in April 1998 at the HP Openview University Association Workshop.....   | 222 |
| 1.  | Claim 1.....  | 223 |
| 2.  | Claim 2.....  | 228 |
| 3.  | Claim 3.....  | 228 |
| 4.  | Claim 6.....  | 229 |
| 5.  | Claim 7.....  | 230 |
| 6.  | Claim 8.....  | 232 |
| 7.  | Claim 9.....  | 232 |
| 8.  | Claim 13.....   | 232 |
| 9.  | Claim 14.....   | 235 |
| 10. | Claim 15.....   | 236 |
| 11. | Claim 18.....   | 236 |
| Q.  | Anticipation by and/or Obviousness in View of <i>Distributed Policy Based Management Enabling Policy Adaptation on Monitoring Using Active Network Technology</i> , by Yoshihara, et al., published in October 2001 at the 12th International Workshop on Distributed Systems. .... | 236 |
| 1.  | Claim 1.....  | 236 |
| 2.  | Claim 2.....  | 240 |
| 3.  | Claim 3.....  | 241 |
| 4.  | Claim 6.....  | 241 |
| 5.  | Claim 7.....  | 242 |
| 6.  | Claim 8.....  | 243 |
| 7.  | Claim 9.....  | 243 |
| 8.  | Claim 13.....   | 244 |
| 9.  | Claim 14.....   | 246 |
| 10. | Claim 15.....   | 246 |
| 11. | Claim 18.....   | 247 |
| R.  | Anticipation by and/or Obviousness in View of Cisco NetRanger, published by Wheelgroup/Cisco by August 1997 at the latest. ....   | 247 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 1.  | Claim 1.....   | 247 |
| 2.  | Claim 2.....   | 251 |
| 3.  | Claim 3.....   | 255 |
| 4.  | Claim 6.....   | 256 |
| 5.  | Claim 7.....   | 257 |
| 6.  | Claim 8.....   | 259 |
| 7.  | Claim 9.....   | 259 |
| 8.  | Claim 13.....  | 259 |
| 9.  | Claim 14.....  | 262 |
| 10. | Claim 15.....  | 262 |
| 11. | Claim 18.....  | 262 |
| S.  | Anticipation by and/or Obviousness in View of the Cisco Catalyst 5000 Switch Family, marketed by Cisco by June 1995 at the latest..... | 262 |
| 1.  | Claim 1.....   | 263 |
| 2.  | Claim 2.....   | 266 |
| 3.  | Claim 6.....   | 267 |
| 4.  | Claim 7.....   | 267 |
| 5.  | Claim 8.....   | 269 |
| 6.  | Claim 9.....   | 269 |
| 7.  | Claim 13.....  | 270 |
| 8.  | Claim 14.....  | 273 |
| 9.  | Claim 15.....  | 273 |
| 10. | Claim 18.....  | 274 |
| T.  | Anticipation by and/or Obviousness in View of [REDACTED].....  | 274 |
| 1.  | Claim 1.....   | 274 |
| 2.  | Claim 2.....   | 277 |
| 3.  | Claim 3.....   | 278 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 4.  | Claim 6.....   | 278 |
| 5.  | Claim 7.....   | 279 |
| 6.  | Claim 8.....   | 280 |
| 7.  | Claim 9.....   | 281 |
| 8.  | Claim 13.....  | 281 |
| 9.  | Claim 14.....  | 285 |
| 10. | Claim 15.....  | 285 |
| 11. | Claim 18.....  | 285 |
| U.  | Anticipation by and/or Obviousness in View of Cisco Receive ACL<br>("rACL"), marketed by Cisco by April 22, 2002 at the latest. ....             | 285 |
| 1.  | Claim 1.....   | 285 |
| 2.  | Claim 2.....   | 294 |
| 3.  | Claim 3.....   | 296 |
| 4.  | Claim 6.....   | 297 |
| 5.  | Claim 7.....   | 298 |
| 6.  | Claim 8.....   | 300 |
| 7.  | Claim 9.....   | 300 |
| 8.  | Claim 13.....  | 300 |
| 9.  | Claim 14.....  | 304 |
| 10. | Claim 15.....  | 304 |
| 11. | Claim 18.....  | 304 |
| V.  | Anticipation by and/or Obviousness in View of the IBM xSeries Server<br>Software Rejuvenation Agent, marketed by IBM by 2000 at the latest. .... | 304 |
| 1.  | Claim 1.....   | 305 |
| 2.  | Claim 2.....   | 311 |
| 3.  | Claim 3.....   | 312 |
| 4.  | Claim 6.....   | 313 |
| 5.  | Claim 7.....   | 314 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 6.  | Claim 8.....   | 317 |
| 7.  | Claim 9.....   | 317 |
| 8.  | Claim 13.....  | 317 |
| 9.  | Claim 14.....  | 320 |
| 10. | Claim 15.....  | 320 |
| 11. | Claim 18.....  | 321 |
| W.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,826,150, filed 10/2/2001 and issued to Bhattacharya, et al. on 11/30/2004. .... | 321 |
| 1.  | Claim 1.....   | 321 |
| 2.  | Claim 2.....   | 325 |
| 3.  | Claim 3.....   | 326 |
| 4.  | Claim 6.....   | 327 |
| 5.  | Claim 7.....   | 327 |
| 6.  | Claim 8.....   | 329 |
| 7.  | Claim 9.....   | 329 |
| 8.  | Claim 13.....  | 330 |
| 9.  | Claim 14.....  | 333 |
| 10. | Claim 15.....  | 333 |
| 11. | Claim 18.....  | 333 |
| X.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,219,719, filed on 1/22/1997 and issued to Graf on 4/17/2001.....                | 333 |
| 1.  | Claim 1.....   | 334 |
| 2.  | Claim 2.....   | 339 |
| 3.  | Claim 3.....   | 340 |
| 4.  | Claim 6.....   | 341 |
| 5.  | Claim 7.....   | 342 |
| 6.  | Claim 8.....   | 344 |
| 7.  | Claim 9.....   | 344 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 8.  | Claim 13.....   | 344 |
| 9.  | Claim 14.....   | 347 |
| 10. | Claim 15.....   | 347 |
| 11. | Claim 18.....   | 348 |
| Y.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,243,667, filed 5/28/1996 and issued to Kerr, et al. on 6/5/2001..... | 348 |
| 1.  | Claim 1.....  | 348 |
| 2.  | Claim 2.....  | 352 |
| 3.  | Claim 3.....  | 352 |
| 4.  | Claim 6.....  | 352 |
| 5.  | Claim 7.....  | 353 |
| 6.  | Claim 8.....  | 355 |
| 7.  | Claim 9.....  | 355 |
| 8.  | Claim 13.....   | 355 |
| 9.  | Claim 14.....   | 357 |
| 10. | Claim 15.....   | 358 |
| 11. | Claim 18.....   | 358 |
| Z.  | Anticipation by and/or Obviousness in View of Cisco NetFlow, marketed by Cisco by 1995 at the latest. ....                                    | 358 |
| 1.  | Claim 1.....  | 358 |
| 2.  | Claim 2.....  | 363 |
| 3.  | Claim 3.....  | 363 |
| 4.  | Claim 6.....  | 364 |
| 5.  | Claim 7.....  | 364 |
| 6.  | Claim 8.....  | 366 |
| 7.  | Claim 9.....  | 367 |
| 8.  | Claim 13.....   | 367 |
| 9.  | Claim 14.....   | 369 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|            |   |            |
|------------|---|------------|
| 10.        | Claim 15.....   | 370        |
| 11.        | Claim 18.....   | 370        |
| <b>IX.</b> | <b>PRIOR ART COMBINATIONS RENDERING THE '659 PATENT<br/>OBVIOUS .....</b>                     | <b>370</b> |
| A.         | Turek in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek .....               | 370        |
| 1.         | Motivation to Combine Turek with Buchanan, Cantrill, and/or<br>Berry .....                    | 370        |
| 2.         | Motivation to Combine Turek with RFC 2328 and/or Afek.....                                    | 372        |
| 3.         | Claim 1 .....   | 372        |
| 4.         | Claim 2.....  | 377        |
| 5.         | Claim 3.....  | 380        |
| 6.         | Claim 6.....  | 380        |
| 7.         | Claim 7.....  | 381        |
| 8.         | Claim 8.....  | 383        |
| 9.         | Claim 9.....  | 383        |
| 10.        | Claim 13.....   | 384        |
| 11.        | Claim 14.....   | 386        |
| 12.        | Claim 15.....   | 386        |
| 13.        | Claim 18.....   | 386        |
| B.         | Goldman in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek and/or Turek..... | 386        |
| 1.         | Motivation to Combine Goldman with Buchanan, Cantrill, and/or<br>Berry .....                  | 386        |
| 2.         | Motivation to Combine Goldman with RFC 2328 and/or Afek .....                                 | 387        |
| 3.         | Motivation to Combine Goldman with Turek .....  | 388        |
| 4.         | Claim 1 .....   | 389        |
| 5.         | Claim 2.....  | 393        |
| 6.         | Claim 3.....  | 396        |
| 7.         | Claim 6.....  | 396        |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 8.  | Claim 7.....  | 398 |
| 9.  | Claim 8.....  | 399 |
| 10. | Claim 9.....  | 400 |
| 11. | Claim 13.....   | 400 |
| 12. | Claim 14.....   | 402 |
| 13. | Claim 15.....   | 402 |
| 14. | Claim 18.....   | 402 |
| C.  | Pandya in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek .....            | 403 |
| 1.  | Motivation to Combine Pandya with Buchanan, Cantrill, and/or<br>Berry .....                 | 403 |
| 2.  | Motivation to Combine Pandya with RFC 2328 and/or Afek .....                                | 404 |
| 3.  | Claim 1.....  | 404 |
| 4.  | Claim 2.....  | 409 |
| 5.  | Claim 3.....  | 412 |
| 6.  | Claim 6.....  | 412 |
| 7.  | Claim 7.....  | 413 |
| 8.  | Claim 8.....  | 415 |
| 9.  | Claim 9.....  | 415 |
| 10. | Claim 13.....   | 416 |
| 11. | Claim 14.....   | 418 |
| 12. | Claim 15.....   | 418 |
| 13. | Claim 18.....   | 418 |
| D.  | Douik in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek and/or Turek..... | 418 |
| 1.  | Motivation to Combine Douik with Buchanan, Cantrill, and/or<br>Berry .....                  | 418 |
| 2.  | Motivation to Combine Douik with RFC 2328 and/or Afek .....                                 | 419 |
| 3.  | Motivation to Combine Douik with Turek .....  | 420 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 4.  | Claim 1.....   | 421 |
| 5.  | Claim 2.....   | 425 |
| 6.  | Claim 3.....   | 428 |
| 7.  | Claim 6.....   | 428 |
| 8.  | Claim 7.....   | 430 |
| 9.  | Claim 8.....   | 431 |
| 10. | Claim 9.....   | 432 |
| 11. | Claim 13.....  | 432 |
| 12. | Claim 14.....  | 434 |
| 13. | Claim 15.....  | 434 |
| 14. | Claim 18.....  | 434 |
| E.  | Natarajan in combination with Cantrill/Berry and/or RFC 2328/Afek<br>and/or Turek..... | 435 |
| 1.  | Motivation to Combine Natarajan with Cantrill, and/or Berry.....                       | 435 |
| 2.  | Motivation to Combine Natarajan with RFC 2328 and/or Afek.....                         | 436 |
| 3.  | Motivation to Combine Natarajan with Turek.....  | 436 |
| 4.  | Claim 1.....   | 437 |
| 5.  | Claim 2.....   | 442 |
| 6.  | Claim 3.....   | 443 |
| 7.  | Claim 6.....   | 443 |
| 8.  | Claim 7.....   | 445 |
| 9.  | Claim 8.....   | 446 |
| 10. | Claim 9.....   | 447 |
| 11. | Claim 13.....  | 447 |
| 12. | Claim 14.....  | 449 |
| 13. | Claim 15.....  | 449 |
| 14. | Claim 18.....  | 450 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| F.  | Bonnell in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek.....  | 450 |
| 1.  | Motivation to Combine Bonnell with Buchanan, Cantrill, and/or Berry .....                   | 450 |
| 2.  | Motivation to Combine Bonnell with RFC 2328 and/or Afek .....                               | 451 |
| 3.  | Motivation to Combine Bonnell with Turek.....   | 452 |
| 4.  | Claim 1.....  | 453 |
| 5.  | Claim 2.....  | 457 |
| 6.  | Claim 3.....  | 460 |
| 7.  | Claim 6.....  | 460 |
| 8.  | Claim 7.....  | 462 |
| 9.  | Claim 8.....  | 463 |
| 10. | Claim 9.....  | 464 |
| 11. | Claim 13.....   | 464 |
| 12. | Claim 14.....   | 466 |
| 13. | Claim 15.....   | 466 |
| 14. | Claim 18.....   | 466 |
| G.  | Boukobza in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek..... | 467 |
| 1.  | Motivation to Combine Boukobza with Buchanan, Cantrill, and/or Berry .....                  | 467 |
| 2.  | Motivation to Combine Boukobza with RFC 2328 and/or Afek.....                               | 468 |
| 3.  | Motivation to Combine Boukobza with Turek .....   | 468 |
| 4.  | Claim 1.....  | 469 |
| 5.  | Claim 2.....  | 474 |
| 6.  | Claim 3.....  | 477 |
| 7.  | Claim 6.....  | 477 |
| 8.  | Claim 7.....  | 479 |
| 9.  | Claim 8.....  | 480 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 10. | Claim 9.....  | 481 |
| 11. | Claim 13.....   | 481 |
| 12. | Claim 14.....   | 483 |
| 13. | Claim 15.....   | 483 |
| 14. | Claim 18.....   | 483 |
| H.  | Hellerstein in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek and/or Turek..... | 484 |
| 1.  | Motivation to Combine Hellerstein with Buchanan, Cantrill, and/or<br>Berry .....                  | 484 |
| 2.  | Motivation to Combine Hellerstein with RFC 2328 and/or Afek.....                                  | 485 |
| 3.  | Motivation to Combine Hellenstein with Turek .....  | 486 |
| 4.  | Claim 1 .....   | 486 |
| 5.  | Claim 2.....  | 491 |
| 6.  | Claim 3.....  | 494 |
| 7.  | Claim 6.....  | 494 |
| 8.  | Claim 7.....  | 496 |
| 9.  | Claim 8.....  | 497 |
| 10. | Claim 9.....  | 498 |
| 11. | Claim 13.....   | 498 |
| 12. | Claim 14.....   | 500 |
| 13. | Claim 15.....   | 500 |
| 14. | Claim 18.....   | 501 |
| I.  | Boudaoud in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek and/or Turek.....    | 501 |
| 1.  | Motivation to Combine Boudaoud with Buchanan, Cantrill, and/or<br>Berry .....                     | 501 |
| 2.  | Motivation to Combine Boudaoud with RFC 2328 and/or Afek.....                                     | 502 |
| 3.  | Motivation to Combine Boudaoud with Turek.....  | 503 |
| 4.  | Claim 1 .....   | 503 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 5.  | Claim 2.....   | 508 |
| 6.  | Claim 3.....   | 511 |
| 7.  | Claim 6.....   | 511 |
| 8.  | Claim 7.....   | 513 |
| 9.  | Claim 8.....   | 514 |
| 10. | Claim 9.....   | 515 |
| 11. | Claim 13.....  | 515 |
| 12. | Claim 14.....  | 517 |
| 13. | Claim 15.....  | 517 |
| 14. | Claim 18.....  | 517 |
| J.  | Kasteleijn in combination with Cantrill/Berry and/or RFC 2328 and/or<br>Lindskog .....         | 518 |
| 1.  | Motivation to Combine Kasteleijn with Cantrill, and/or Berry .....                             | 518 |
| 2.  | Motivation to Combine Kasteleijn with RFC 2328 .....   | 519 |
| 3.  | Motivation to Combine Kasteleijn with Lindskog.....  | 519 |
| 4.  | Claim 1 .....  | 520 |
| 5.  | Claim 2.....   | 525 |
| 6.  | Claim 3.....   | 526 |
| 7.  | Claim 6.....   | 526 |
| 8.  | Claim 7.....   | 527 |
| 9.  | Claim 8.....   | 529 |
| 10. | Claim 9.....   | 529 |
| 11. | Claim 13.....  | 530 |
| 12. | Claim 14.....  | 532 |
| 13. | Claim 15.....  | 532 |
| 14. | Claim 18.....  | 532 |
| K.  | Da Rocha in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek and/or Turek..... | 532 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 1.  | Motivation to Combine Da Rocha with Buchanan, Cantrill, and/or Berry .....                    | 533 |
| 2.  | Motivation to Combine Da Rocha with RFC 2328 and/or Afek .....                                | 534 |
| 3.  | Motivation to Combine da Rocha with Turek .....   | 534 |
| 4.  | Claim 1 .....   | 535 |
| 5.  | Claim 2 .....   | 540 |
| 6.  | Claim 3 .....   | 543 |
| 7.  | Claim 6 .....   | 543 |
| 8.  | Claim 7 .....   | 545 |
| 9.  | Claim 8 .....   | 546 |
| 10. | Claim 9 .....   | 547 |
| 11. | Claim 13 .....  | 547 |
| 12. | Claim 14 .....  | 549 |
| 13. | Claim 15 .....  | 549 |
| 14. | Claim 18 .....  | 549 |
| L.  | Castelli in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek ..... | 550 |
| 1.  | Motivation to Combine Castelli with Buchanan, Cantrill, and/or Berry .....                    | 550 |
| 2.  | Motivation to Combine Castelli with RFC 2328 and/or Afek .....                                | 551 |
| 3.  | Motivation to Combine Castelli with Turek .....   | 551 |
| 4.  | Claim 1 .....   | 552 |
| 5.  | Claim 2 .....   | 557 |
| 6.  | Claim 3 .....   | 560 |
| 7.  | Claim 6 .....   | 560 |
| 8.  | Claim 7 .....   | 562 |
| 9.  | Claim 8 .....   | 563 |
| 10. | Claim 9 .....   | 564 |
| 11. | Claim 13 .....  | 564 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 12. | Claim 14.....  | 566 |
| 13. | Claim 15.....  | 566 |
| 14. | Claim 18.....  | 566 |
| M.  | Conti in combination with Kasteleijn/Turek/Goldman and/or RFC<br>2328/Afek and/or Turek.....     | 567 |
| 1.  | Motivation to Combine Conti with Buchanan, Cantrill, and/or<br>Berry .....                       | 567 |
| 2.  | Motivation to Combine Conti with RFC 2328 and/or Afek .....                                      | 568 |
| 3.  | Motivation to Combine Conti with Turek .....   | 568 |
| 4.  | Claim 1 .....  | 569 |
| 5.  | Claim 2.....   | 574 |
| 6.  | Claim 3.....   | 577 |
| 7.  | Claim 6.....   | 577 |
| 8.  | Claim 7.....   | 579 |
| 9.  | Claim 8.....   | 580 |
| 10. | Claim 9.....   | 580 |
| 11. | Claim 13.....  | 581 |
| 12. | Claim 14.....  | 583 |
| 13. | Claim 15.....  | 583 |
| 14. | Claim 18.....  | 583 |
| N.  | NetRanger in combination with Kasteleijn/Turek/Goldman and/or RFC<br>2328/Afek and/or Turek..... | 583 |
| 1.  | Motivation to Combine NetRanger with Buchanan, Cantrill,<br>and/or Berry .....                   | 584 |
| 2.  | Motivation to Combine NetRanger with RFC 2328 and/or Afek.....                                   | 585 |
| 3.  | Motivation to Combine NetRanger with Turek.....  | 585 |
| 4.  | Claim 1 .....  | 586 |
| 5.  | Claim 2.....   | 591 |
| 6.  | Claim 3.....   | 593 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 7.  | Claim 6.....   | 594 |
| 8.  | Claim 7.....   | 595 |
| 9.  | Claim 8.....   | 597 |
| 10. | Claim 9.....   | 597 |
| 11. | Claim 13.....  | 598 |
| 12. | Claim 14.....  | 600 |
| 13. | Claim 15.....  | 600 |
| 14. | Claim 18.....  | 600 |
| O.  | NSMIA in combination with Turek, Buchanan/Cantrill/Berry, and/or RFC 2328/Afek and/or Turek..... | 600 |
| 1.  | Motivation to Combine NSMIA with Buchanan, Cantrill, and/or Berry .....                          | 601 |
| 2.  | Motivation to Combine NSMIA with RFC 2328 and/or Afek .....                                      | 602 |
| 3.  | Motivation to Combine NSMIA with Turek .....   | 602 |
| 4.  | Claim 1.....   | 603 |
| 5.  | Claim 2.....   | 608 |
| 6.  | Claim 3.....   | 610 |
| 7.  | Claim 6.....   | 611 |
| 8.  | Claim 7.....   | 612 |
| 9.  | Claim 8.....   | 614 |
| 10. | Claim 9.....   | 614 |
| 11. | Claim 13.....  | 615 |
| 12. | Claim 14.....  | 617 |
| 13. | Claim 15.....  | 617 |
| 14. | Claim 18.....  | 617 |
| P.  | [REDACTED] in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek .....  | 617 |
| 1.  | Motivation to Combine [REDACTED] with Buchanan, Cantrill, and/or Berry .....                     | 617 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 2.  | Motivation to Combine [REDACTED] with RFC 2328 and/or Afek.....   | 619 |
| 3.  | Motivation to Combine [REDACTED] with Turek.....  | 619 |
| 4.  | Claim 1.....  | 620 |
| 5.  | Claim 2.....  | 626 |
| 6.  | Claim 3.....  | 628 |
| 7.  | Claim 6.....  | 629 |
| 8.  | Claim 7.....  | 631 |
| 9.  | Claim 8.....  | 632 |
| 10. | Claim 9.....  | 633 |
| 11. | Claim 13.....   | 633 |
| 12. | Claim 14.....   | 635 |
| 13. | Claim 15.....   | 636 |
| 14. | Claim 18.....   | 636 |
| Q.  | Cisco Catalyst 5000 in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek..... | 636 |
| 1.  | Motivation to Combine Cisco Catalyst 5000R with Buchanan, Cantrill, and/or Berry.....                   | 636 |
| 2.  | Motivation to Combine Cisco Catalyst 5000 with RFC 2328 and/or Afek.....                                | 637 |
| 3.  | Motivation to Combine Cisco Catalyst 5000 with Turek.....   | 638 |
| 4.  | Claim 1.....  | 639 |
| 5.  | Claim 2.....  | 644 |
| 6.  | Claim 3.....  | 646 |
| 7.  | Claim 6.....  | 646 |
| 8.  | Claim 7.....  | 648 |
| 9.  | Claim 8.....  | 649 |
| 10. | Claim 9.....  | 650 |
| 11. | Claim 13.....   | 650 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 12. | Claim 14.....   | 652 |
| 13. | Claim 15.....   | 652 |
| 14. | Claim 18.....   | 653 |
| R.  | Cisco rACL in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek.....                                      | 653 |
| 1.  | Motivation to Combine Cisco rACLR with Buchanan, Cantrill, and/or Berry .....   | 653 |
| 2.  | Motivation to Combine Cisco rACL with RFC 2328 and/or Afek.....   | 654 |
| 3.  | Motivation to Combine Cisco rACL with Turek.....  | 655 |
| 4.  | Claim 1 .....   | 655 |
| 5.  | Claim 2.....  | 660 |
| 6.  | Claim 3.....  | 663 |
| 7.  | Claim 6.....  | 663 |
| 8.  | Claim 7.....  | 665 |
| 9.  | Claim 8.....  | 666 |
| 10. | Claim 9.....  | 667 |
| 11. | Claim 13.....   | 667 |
| 12. | Claim 14.....   | 669 |
| 13. | Claim 15.....   | 669 |
| 14. | Claim 18.....   | 670 |
| S.  | IBM xSeries Server Software Rejuvenation Agent in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek ..... | 670 |
| 1.  | Motivation to Combine IBM xSeries Server Software Rejuvenation AgentR with Buchanan, Cantrill, and/or Berry .....                   | 670 |
| 2.  | Motivation to Combine IBM xSeries Server Software Rejuvenation Agent with RFC 2328 and/or Afek.....                                 | 671 |
| 3.  | Motivation to Combine IBM xSeries Server Software Rejuvenation Agent with Turek .....   | 672 |
| 4.  | Claim 1 .....   | 673 |
| 5.  | Claim 2.....  | 678 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 6.  | Claim 3.....   | 680 |
| 7.  | Claim 6.....   | 681 |
| 8.  | Claim 7.....   | 683 |
| 9.  | Claim 8.....   | 684 |
| 10. | Claim 9.....   | 685 |
| 11. | Claim 13.....  | 685 |
| 12. | Claim 14.....  | 687 |
| 13. | Claim 15.....  | 688 |
| 14. | Claim 18.....  | 688 |
| T.  | INCA in combination with Buchanan/Cantrill/Berry and/or RFC<br>2328/Afek and/or Turek..... | 689 |
| 1.  | Motivation to Combine INCA with Buchanan, Cantrill, and/or<br>Berry .....                  | 689 |
| 2.  | Motivation to Combine INCA with RFC 2328 and/or Afek .....                                 | 690 |
| 3.  | Motivation to Combine INCA with Turek.....   | 690 |
| 4.  | Claim 1 .....  | 691 |
| 5.  | Claim 2.....   | 696 |
| 6.  | Claim 3.....   | 699 |
| 7.  | Claim 6.....   | 699 |
| 8.  | Claim 7.....   | 700 |
| 9.  | Claim 8.....   | 702 |
| 10. | Claim 9.....   | 702 |
| 11. | Claim 13.....  | 703 |
| 12. | Claim 14.....  | 705 |
| 13. | Claim 15.....  | 705 |
| 14. | Claim 18.....  | 705 |
| U.  | Yoshihara in combination with Cantrill/Berry and/or RFC 2328/Afek<br>and/or Turek.....     | 705 |
| 1.  | Motivation to Combine Yoshihara with Cantrill and/or Berry .....                           | 705 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 2.  | Motivation to Combine Yoshihara with RFC 2328 and/or Afek.....                              | 706 |
| 3.  | Motivation to Combine Yoshihara with Turek.....   | 707 |
| 4.  | Claim 1.....  | 708 |
| 5.  | Claim 2.....  | 712 |
| 6.  | Claim 3.....  | 713 |
| 7.  | Claim 6.....  | 713 |
| 8.  | Claim 7.....  | 715 |
| 9.  | Claim 8.....  | 717 |
| 10. | Claim 9.....  | 717 |
| 11. | Claim 13.....   | 717 |
| 12. | Claim 14.....   | 719 |
| 13. | Claim 15.....   | 720 |
| 14. | Claim 18.....   | 720 |
| V.  | Cisco Catalyst 5000 Switch Family in combination with NetRanger and Goldman.....            | 720 |
| 1.  | Motivation to Combine Cisco Catalyst 5000 Switch with NetRanger and/or Goldman .....        | 720 |
| 2.  | Motivation to Combine Cisco NetRanger with Buchanan, Cantrill, and/or Berry .....           | 721 |
| 3.  | Motivation to Combine Cisco NetRanger with RFC 2328 and/or Afek.....                        | 722 |
| W.  | Lindskog in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek..... | 722 |
| 1.  | Motivation to Combine Lindskog with Buchanan, Cantrill, and/or Berry .....                  | 722 |
| 2.  | Motivation to Combine Lindskog with RFC 2328 and/or Afek.....                               | 723 |
| 3.  | Motivation to Combine Lindskog with Turek .....   | 724 |
| 4.  | Claim 1.....  | 725 |
| 5.  | Claim 2.....  | 730 |
| 6.  | Claim 3.....  | 732 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|           |  |            |
|-----------|--|------------|
| 7.        | Claim 6.....   | 733        |
| 8.        | Claim 7.....   | 734        |
| 9.        | Claim 8.....   | 736        |
| 10.       | Claim 9.....   | 736        |
| 11.       | Claim 13.....  | 737        |
| 12.       | Claim 14.....  | 739        |
| 13.       | Claim 15.....  | 739        |
| 14.       | Claim 18.....  | 739        |
| <b>X.</b> | <b>ANTICIPATORY PRIOR ART FOR THE '730 PATENT .....</b>  | <b>740</b> |
| A.        | Anticipation by and/or Obviousness in View of United States Patent No. 6,460,070, filed 6/3/1998 and issued to Turek, et al. on 10/2/2002..... | 740        |
| 1.        | Claim 1.....   | 740        |
| 2.        | Claim 2.....   | 750        |
| 3.        | Claim 3.....   | 750        |
| 4.        | Claim 4.....   | 752        |
| 5.        | Claim 6.....   | 752        |
| 6.        | Claim 7.....   | 753        |
| 7.        | Claim 10.....  | 756        |
| 8.        | Claim 11.....  | 756        |
| 9.        | Claim 12.....  | 757        |
| B.        | Anticipation by and/or Obviousness in View of Great Britain Patent No. 2 363 284, issued to Goldman, et al. on 12/12/2001.....                 | 758        |
| 1.        | Claim 1.....   | 758        |
| 2.        | Claim 2.....   | 763        |
| 3.        | Claim 3.....   | 764        |
| 4.        | Claim 4.....   | 765        |
| 5.        | Claim 6.....   | 765        |
| 6.        | Claim 7.....   | 766        |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 7.  | Claim 10.....   | 768 |
| 8.  | Claim 11.....   | 769 |
| 9.  | Claim 12.....   | 769 |
| 10. | Claim 16.....   | 770 |
| 11. | Claim 17.....   | 771 |
| 12. | Claim 18.....   | 771 |
| 13. | Claim 19.....   | 772 |
| 14. | Claim 21.....   | 772 |
| 15. | Claim 22.....   | 773 |
| 16. | Claim 24.....   | 773 |
| 17. | Claim 26.....   | 774 |
| 18. | Claim 29.....   | 774 |
| 19. | Claim 30.....   | 775 |
| 20. | Claim 31.....   | 777 |
| 21. | Claim 32.....   | 777 |
| 22. | Claim 33.....   | 778 |
| 23. | Claim 34.....   | 778 |
| 24. | Claim 36.....   | 779 |
| C.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,671,724, filed 3/21/2000 and issued to Pandya, et al. on 12/30/2003. .... | 779 |
| 1.  | Claim 1.....  | 779 |
| 2.  | Claim 2.....  | 786 |
| 3.  | Claim 3.....  | 788 |
| 4.  | Claim 4.....  | 789 |
| 5.  | Claim 6.....  | 789 |
| 6.  | Claim 7.....  | 790 |
| 7.  | Claim 10.....   | 792 |
| 8.  | Claim 11.....   | 792 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 9.  | Claim 12.....  | 793 |
| 10. | Claim 16.....  | 793 |
| 11. | Claim 17.....  | 794 |
| 12. | Claim 18.....  | 794 |
| 13. | Claim 19.....  | 794 |
| 14. | Claim 21.....  | 795 |
| 15. | Claim 22.....  | 796 |
| 16. | Claim 24.....  | 796 |
| 17. | Claim 26.....  | 796 |
| 18. | Claim 29.....  | 797 |
| 19. | Claim 30.....  | 797 |
| 20. | Claim 31.....  | 800 |
| 21. | Claim 32.....  | 800 |
| 22. | Claim 33.....  | 800 |
| 23. | Claim 34.....  | 801 |
| 24. | Claim 36.....  | 801 |
| D.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,012,152, issued to Douik, et al. on 1/4/2000. .... | 801 |
| 1.  | Claim 1.....   | 802 |
| 2.  | Claim 2.....   | 807 |
| 3.  | Claim 3.....   | 807 |
| 4.  | Claim 4.....   | 808 |
| 5.  | Claim 6.....   | 809 |
| 6.  | Claim 7.....   | 809 |
| 7.  | Claim 10.....  | 812 |
| 8.  | Claim 11.....  | 812 |
| 9.  | Claim 12.....  | 812 |
| 10. | Claim 16.....  | 813 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 11. | Claim 17.....  | 813 |
| 12. | Claim 18.....  | 814 |
| 13. | Claim 19.....  | 815 |
| 14. | Claim 21.....  | 815 |
| 15. | Claim 22.....  | 816 |
| 16. | Claim 24.....  | 816 |
| 17. | Claim 26.....  | 816 |
| 18. | Claim 29.....  | 817 |
| 19. | Claim 30.....  | 817 |
| 20. | Claim 31.....  | 820 |
| 21. | Claim 32.....  | 820 |
| 22. | Claim 33.....  | 820 |
| 23. | Claim 34.....  | 821 |
| 24. | Claim 36.....  | 821 |
| E.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,584,502, filed 6/29/1999 and issued to Natarajan, et al. on 6/24/2003..... | 821 |
| 1.  | Claim 1.....   | 822 |
| 2.  | Claim 2.....   | 826 |
| 3.  | Claim 3.....   | 827 |
| 4.  | Claim 4.....   | 827 |
| 5.  | Claim 6.....   | 828 |
| 6.  | Claim 7.....   | 828 |
| 7.  | Claim 10.....  | 830 |
| 8.  | Claim 11.....  | 831 |
| 9.  | Claim 12.....  | 831 |
| 10. | Claim 16.....  | 832 |
| 11. | Claim 17.....  | 832 |
| 12. | Claim 18.....  | 833 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 13. | Claim 19.....   | 833 |
| 14. | Claim 21.....   | 834 |
| 15. | Claim 22.....   | 834 |
| 16. | Claim 24.....   | 834 |
| 17. | Claim 26.....   | 835 |
| 18. | Claim 29.....   | 835 |
| 19. | Claim 30.....   | 836 |
| 20. | Claim 31.....   | 838 |
| 21. | Claim 32.....   | 838 |
| 22. | Claim 33.....   | 839 |
| 23. | Claim 34.....   | 839 |
| 24. | Claim 36.....   | 840 |
| F.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>5,655,081, issued to Bonnell, et al. on 8/5/1997..... | 840 |
| 1.  | Claim 1.....  | 840 |
| 2.  | Claim 2.....  | 846 |
| 3.  | Claim 3.....  | 846 |
| 4.  | Claim 4.....  | 847 |
| 5.  | Claim 6.....  | 847 |
| 6.  | Claim 7.....  | 847 |
| 7.  | Claim 10.....   | 850 |
| 8.  | Claim 11.....   | 850 |
| 9.  | Claim 12.....   | 851 |
| 10. | Claim 16.....   | 851 |
| 11. | Claim 17.....   | 851 |
| 12. | Claim 18.....   | 852 |
| 13. | Claim 19.....   | 852 |
| 14. | Claim 21.....   | 853 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 15. | Claim 22.....   | 853 |
| 16. | Claim 24.....   | 854 |
| 17. | Claim 26.....   | 854 |
| 18. | Claim 29.....   | 854 |
| 19. | Claim 30.....   | 855 |
| 20. | Claim 31.....   | 857 |
| 21. | Claim 32.....   | 857 |
| 22. | Claim 33.....   | 858 |
| 23. | Claim 34.....   | 858 |
| 24. | Claim 36.....   | 859 |
| G.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,122,664, issued to Boukobza, et al. on 9/19/2000. .... | 859 |
| 1.  | Claim 1.....  | 859 |
| 2.  | Claim 2.....  | 864 |
| 3.  | Claim 3.....  | 864 |
| 4.  | Claim 4.....  | 865 |
| 5.  | Claim 6.....  | 866 |
| 6.  | Claim 7.....  | 866 |
| 7.  | Claim 10.....   | 868 |
| 8.  | Claim 11.....   | 868 |
| 9.  | Claim 12.....   | 869 |
| 10. | Claim 16.....   | 869 |
| 11. | Claim 17.....   | 870 |
| 12. | Claim 18.....   | 871 |
| 13. | Claim 19.....   | 871 |
| 14. | Claim 21.....   | 871 |
| 15. | Claim 22.....   | 872 |
| 16. | Claim 24.....   | 873 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 17. | Claim 26.....  | 873 |
| 18. | Claim 29.....  | 873 |
| 19. | Claim 30.....  | 874 |
| 20. | Claim 31.....  | 876 |
| 21. | Claim 32.....  | 876 |
| 22. | Claim 33.....  | 877 |
| 23. | Claim 34.....  | 877 |
| 24. | Claim 36.....  | 878 |
| H.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,665,262, filed 2/16/1999 and issued to Linskog, et al. on 12/16/2003..... | 878 |
| 1.  | Claim 1.....   | 878 |
| 2.  | Claim 2.....   | 884 |
| 3.  | Claim 3.....   | 884 |
| 4.  | Claim 4.....   | 885 |
| 5.  | Claim 6.....   | 886 |
| 6.  | Claim 7.....   | 886 |
| 7.  | Claim 10.....  | 889 |
| 8.  | Claim 11.....  | 889 |
| 9.  | Claim 12.....  | 889 |
| 10. | Claim 16.....  | 890 |
| 11. | Claim 17.....  | 891 |
| 12. | Claim 18.....  | 891 |
| 13. | Claim 19.....  | 892 |
| 14. | Claim 21.....  | 892 |
| 15. | Claim 22.....  | 893 |
| 16. | Claim 24.....  | 893 |
| 17. | Claim 26.....  | 894 |
| 18. | Claim 29.....  | 894 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 19. | Claim 30.....   | 895 |
| 20. | Claim 31.....   | 897 |
| 21. | Claim 32.....   | 897 |
| 22. | Claim 33.....   | 897 |
| 23. | Claim 34.....   | 898 |
| 24. | Claim 36.....   | 898 |
| I.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,675,128, filed 9/30/1999 and issued to Hellerstein, et al. on 1/6/2004..... | 898 |
| 1.  | Claim 1.....  | 899 |
| 2.  | Claim 2.....  | 906 |
| 3.  | Claim 3.....  | 906 |
| 4.  | Claim 4.....  | 907 |
| 5.  | Claim 6.....  | 908 |
| 6.  | Claim 7.....  | 908 |
| 7.  | Claim 10.....   | 910 |
| 8.  | Claim 11.....   | 911 |
| 9.  | Claim 12.....   | 911 |
| 10. | Claim 16.....   | 911 |
| 11. | Claim 17.....   | 912 |
| 12. | Claim 18.....   | 912 |
| 13. | Claim 19.....   | 913 |
| 14. | Claim 21.....   | 913 |
| 15. | Claim 22.....   | 913 |
| 16. | Claim 24.....   | 914 |
| 17. | Claim 26.....   | 914 |
| 18. | Claim 29.....   | 914 |
| 19. | Claim 30.....   | 915 |
| 20. | Claim 31.....   | 917 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 21. | Claim 32.....   | 917 |
| 22. | Claim 33.....   | 917 |
| 23. | Claim 34.....   | 918 |
| 24. | Claim 36.....   | 918 |
| J.  | Anticipation by and/or Obviousness in View of <i>Distributed Management with Mobile Components</i> , by Kasteleijn, et al., published in the May 1999 Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management. .... | 918 |
| 1.  | Claim 1.....  | 919 |
| 2.  | Claim 2.....  | 924 |
| 3.  | Claim 3.....  | 925 |
| 4.  | Claim 4.....  | 925 |
| 5.  | Claim 6.....  | 926 |
| 6.  | Claim 7.....  | 926 |
| 7.  | Claim 10.....   | 928 |
| 8.  | Claim 11.....   | 928 |
| 9.  | Claim 12.....   | 929 |
| 10. | Claim 16.....   | 930 |
| 11. | Claim 17.....   | 930 |
| 12. | Claim 18.....   | 931 |
| 13. | Claim 19.....   | 931 |
| 14. | Claim 21.....   | 931 |
| 15. | Claim 22.....   | 932 |
| 16. | Claim 24.....   | 932 |
| 17. | Claim 26.....   | 932 |
| 18. | Claim 29.....   | 932 |
| 19. | Claim 30.....   | 933 |
| 20. | Claim 31.....   | 935 |
| 21. | Claim 32.....   | 935 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |     |
|-----|---|-----|
| 22. | Claim 33.....   | 936 |
| 23. | Claim 34.....   | 936 |
| 24. | Claim 36.....   | 936 |
| K.  | Anticipation by and/or Obviousness in View of <i>Proactive Management of Computer Networks using Artificial Intelligence Agents and Techniques</i> , by da Rocha, et al., published in the 1997 issue of Integrated Network Management V (Springer). .... | 937 |
| 1.  | Claim 1.....  | 937 |
| 2.  | Claim 2.....  | 945 |
| 3.  | Claim 3.....  | 945 |
| 4.  | Claim 4.....  | 946 |
| 5.  | Claim 6.....  | 946 |
| 6.  | Claim 7.....  | 947 |
| 7.  | Claim 10.....   | 949 |
| 8.  | Claim 11.....   | 950 |
| 9.  | Claim 12.....   | 950 |
| 10. | Claim 16.....   | 951 |
| 11. | Claim 17.....   | 951 |
| 12. | Claim 18.....   | 952 |
| 13. | Claim 19.....   | 953 |
| 14. | Claim 21.....   | 953 |
| 15. | Claim 22.....   | 954 |
| 16. | Claim 24.....   | 955 |
| 17. | Claim 26.....   | 955 |
| 18. | Claim 29.....   | 956 |
| 19. | Claim 30.....   | 957 |
| 20. | Claim 31.....   | 959 |
| 21. | Claim 32.....   | 959 |
| 22. | Claim 33.....   | 960 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |     |
|-----|--|-----|
| 23. | Claim 34.....  | 960 |
| 24. | Claim 36.....  | 961 |
| L.  | Anticipation by and/or Obviousness in View of Proactive Management of<br>Software Aging, by Castelli, et al., published in March 2001 by IBM. .... | 961 |
| 1.  | Claim 1.....   | 961 |
| 2.  | Claim 2.....   | 972 |
| 3.  | Claim 3.....   | 973 |
| 4.  | Claim 4.....   | 974 |
| 5.  | Claim 6.....   | 975 |
| 6.  | Claim 7.....   | 976 |
| 7.  | Claim 10.....  | 981 |
| 8.  | Claim 11.....  | 981 |
| 9.  | Claim 12.....  | 982 |
| 10. | Claim 16.....  | 982 |
| 11. | Claim 17.....  | 983 |
| 12. | Claim 18.....  | 983 |
| 13. | Claim 19.....  | 983 |
| 14. | Claim 21.....  | 984 |
| 15. | Claim 22.....  | 984 |
| 16. | Claim 24.....  | 986 |
| 17. | Claim 26.....  | 986 |
| 18. | Claim 29.....  | 987 |
| 19. | Claim 30.....  | 987 |
| 20. | Claim 31.....  | 989 |
| 21. | Claim 32.....  | 990 |
| 22. | Claim 33.....  | 990 |
| 23. | Claim 34.....  | 991 |
| 24. | Claim 36.....  | 991 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| M.  | Anticipation by and/or Obviousness in View of <i>Network Security Management with Intelligent Agents - A First Step with SLD ("Conti")</i> , published in the HP OpenView University Association Workshop. .... | 991  |
| 1.  | Claim 1.....  | 991  |
| 2.  | Claim 2.....  | 1006 |
| 3.  | Claim 3.....  | 1007 |
| 4.  | Claim 4.....  | 1008 |
| 5.  | Claim 6.....  | 1009 |
| 6.  | Claim 7.....  | 1011 |
| 7.  | Claim 10.....   | 1014 |
| 8.  | Claim 11.....   | 1014 |
| 9.  | Claim 12.....   | 1015 |
| 10. | Claim 13.....   | 1016 |
| 11. | Claim 16.....   | 1016 |
| 12. | Claim 17.....   | 1017 |
| 13. | Claim 18.....   | 1018 |
| 14. | Claim 19.....   | 1019 |
| 15. | Claim 21.....   | 1020 |
| 16. | Claim 22.....   | 1021 |
| 17. | Claim 24.....   | 1021 |
| 18. | Claim 26.....   | 1021 |
| 19. | Claim 29.....   | 1023 |
| 20. | Claim 30.....   | 1024 |
| 21. | Claim 31.....   | 1026 |
| 22. | Claim 32.....   | 1027 |
| 23. | Claim 33.....   | 1027 |
| 24. | Claim 34.....   | 1027 |
| 25. | Claim 36.....   | 1028 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| N.  | Anticipation by and/or Obviousness in View of <i>Network Security Management with Intelligent Agents</i> (“NSMIA”), NOMS 2000. 2000 IEEE/IFIP Network Operations and Management Symposium 'The Networked Planet: Management Beyond 2000. .... | 1028 |
| 1.  | Claim 1.....  | 1028 |
| 2.  | Claim 2.....  | 1041 |
| 3.  | Claim 3.....  | 1042 |
| 4.  | Claim 4.....  | 1044 |
| 5.  | Claim 6.....  | 1044 |
| 6.  | Claim 7.....  | 1045 |
| 7.  | Claim 10.....   | 1048 |
| 8.  | Claim 11.....   | 1048 |
| 9.  | Claim 12.....   | 1049 |
| 10. | Claim 16.....   | 1050 |
| 11. | Claim 17.....   | 1051 |
| 12. | Claim 18.....   | 1051 |
| 13. | Claim 19.....   | 1052 |
| 14. | Claim 21.....   | 1053 |
| 15. | Claim 22.....   | 1053 |
| 16. | Claim 24.....   | 1054 |
| 17. | Claim 26.....   | 1054 |
| 18. | Claim 29.....   | 1055 |
| 19. | Claim 30.....   | 1056 |
| 20. | Claim 31.....   | 1058 |
| 21. | Claim 32.....   | 1059 |
| 22. | Claim 33.....   | 1059 |
| 23. | Claim 34.....   | 1060 |
| 24. | Claim 36.....   | 1060 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| O.  | Anticipation by and/or Obviousness in View of <i>INCA: An Agent-based Network Control Architecture</i> , published in the 1998 International Workshop on Intelligent Agents for Telecommunication Applications. .... | 1060 |
| 1.  | Claim 1.....   | 1060 |
| 2.  | Claim 2.....   | 1066 |
| 3.  | Claim 3.....   | 1066 |
| 4.  | Claim 4.....   | 1068 |
| 5.  | Claim 6.....   | 1068 |
| 6.  | Claim 7.....   | 1069 |
| 7.  | Claim 10.....  | 1071 |
| 8.  | Claim 11.....  | 1071 |
| 9.  | Claim 12.....  | 1072 |
| 10. | Claim 16.....  | 1072 |
| 11. | Claim 17.....  | 1073 |
| 12. | Claim 18.....  | 1074 |
| 13. | Claim 19.....  | 1075 |
| 14. | Claim 21.....  | 1075 |
| 15. | Claim 22.....  | 1076 |
| 16. | Claim 24.....  | 1076 |
| 17. | Claim 26.....  | 1076 |
| 18. | Claim 29.....  | 1077 |
| 19. | Claim 30.....  | 1078 |
| 20. | Claim 31.....  | 1080 |
| 21. | Claim 32.....  | 1080 |
| 22. | Claim 33.....  | 1081 |
| 23. | Claim 34.....  | 1081 |
| 24. | Claim 36.....  | 1082 |
| P.  | Anticipation by and/or Obviousness in View of <i>Distributed Network Security Management Using Intelligent Agents</i> , by Boudaoud, et al.,   |      |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|   |      |
|---|------|
| published in April 1998 at the HP Openview University Association Workshop..... | 1082 |
| 1. Claim 1.....   | 1082 |
| 2. Claim 2.....   | 1089 |
| 3. Claim 3.....   | 1089 |
| 4. Claim 4.....   | 1091 |
| 5. Claim 6.....   | 1091 |
| 6. Claim 7.....   | 1092 |
| 7. Claim 10.....  | 1095 |
| 8. Claim 11.....  | 1095 |
| 9. Claim 12.....  | 1095 |
| 10. Claim 16.....   | 1096 |
| 11. Claim 17.....   | 1096 |
| 12. Claim 18.....   | 1097 |
| 13. Claim 19.....   | 1097 |
| 14. Claim 21.....   | 1098 |
| 15. Claim 22.....   | 1098 |
| 16. Claim 24.....   | 1099 |
| 17. Claim 26.....   | 1099 |
| 18. Claim 29.....   | 1100 |
| 19. Claim 30.....   | 1101 |
| 20. Claim 31.....   | 1103 |
| 21. Claim 32.....   | 1104 |
| 22. Claim 33.....   | 1104 |
| 23. Claim 34.....   | 1105 |
| 24. Claim 36.....   | 1105 |

Q. Anticipation by and/or Obviousness in View of *Distributed Policy Based Management Enabling Policy Adaptation on Monitoring Using Active*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
|     | <i>Network Technology</i> , by Yoshihara, et al., published in October 2001 at the 12th International Workshop on Distributed Systems. .... | 1105 |
| 1.  | Claim 1 .....   | 1105 |
| 2.  | Claim 2 .....   | 1110 |
| 3.  | Claim 3 .....   | 1111 |
| 4.  | Claim 4 .....   | 1111 |
| 5.  | Claim 6 .....   | 1112 |
| 6.  | Claim 7 .....   | 1112 |
| 7.  | Claim 10 .....  | 1114 |
| 8.  | Claim 11 .....  | 1115 |
| 9.  | Claim 12 .....  | 1115 |
| 10. | Claim 16 .....  | 1116 |
| 11. | Claim 17 .....  | 1116 |
| 12. | Claim 18 .....  | 1116 |
| 13. | Claim 19 .....  | 1117 |
| 14. | Claim 21 .....  | 1117 |
| 15. | Claim 22 .....  | 1118 |
| 16. | Claim 24 .....  | 1118 |
| 17. | Claim 26 .....  | 1118 |
| 18. | Claim 29 .....  | 1119 |
| 19. | Claim 30 .....  | 1119 |
| 20. | Claim 31 .....  | 1121 |
| 21. | Claim 32 .....  | 1121 |
| 22. | Claim 33 .....  | 1122 |
| 23. | Claim 34 .....  | 1122 |
| 24. | Claim 36 .....  | 1123 |
| R.  | Anticipation by and/or Obviousness in View of Cisco NetRanger, published by Wheelgroup/Cisco by August 1997 at the latest. ....             | 1123 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 1.  | Claim 1.....  | 1123 |
| 2.  | Claim 2.....  | 1129 |
| 3.  | Claim 3.....  | 1130 |
| 4.  | Claim 4.....  | 1131 |
| 5.  | Claim 6.....  | 1132 |
| 6.  | Claim 7.....  | 1132 |
| 7.  | Claim 10.....   | 1135 |
| 8.  | Claim 11.....   | 1136 |
| 9.  | Claim 12.....   | 1136 |
| 10. | Claim 16.....   | 1136 |
| 11. | Claim 17.....   | 1137 |
| 12. | Claim 18.....   | 1138 |
| 13. | Claim 19.....   | 1138 |
| 14. | Claim 21.....   | 1139 |
| 15. | Claim 22.....   | 1140 |
| 16. | Claim 24.....   | 1140 |
| 17. | Claim 26.....   | 1140 |
| 18. | Claim 29.....   | 1141 |
| 19. | Claim 30.....   | 1141 |
| 20. | Claim 31.....   | 1144 |
| 21. | Claim 32.....   | 1144 |
| 22. | Claim 33.....   | 1144 |
| 23. | Claim 34.....   | 1145 |
| 24. | Claim 36.....   | 1145 |
| S.  | Anticipation by and/or Obviousness in View of the Cisco Catalyst 5000<br>Switch Family, marketed by Cisco by June 1995 at the latest..... | 1145 |
| 1.  | Claim 1.....  | 1146 |
| 2.  | Claim 2.....  | 1154 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 3.  | Claim 3.....   | 1154 |
| 4.  | Claim 4.....   | 1155 |
| 5.  | Claim 6.....   | 1155 |
| 6.  | Claim 7.....   | 1156 |
| 7.  | Claim 10.....  | 1159 |
| 8.  | Claim 11.....  | 1159 |
| 9.  | Claim 12.....  | 1160 |
| 10. | Claim 16.....  | 1161 |
| 11. | Claim 17.....  | 1161 |
| 12. | Claim 18.....  | 1162 |
| 13. | Claim 19.....  | 1162 |
| 14. | Claim 21.....  | 1162 |
| 15. | Claim 22.....  | 1163 |
| 16. | Claim 24.....  | 1164 |
| 17. | Claim 26.....  | 1164 |
| 18. | Claim 29.....  | 1164 |
| 19. | Claim 30.....  | 1164 |
| 20. | Claim 31.....  | 1167 |
| 21. | Claim 32.....  | 1167 |
| 22. | Claim 33.....  | 1168 |
| 23. | Claim 34.....  | 1168 |
| 24. | Claim 36.....  | 1169 |
| T.  | Anticipation by and/or Obviousness in View of [REDACTED]<br>[REDACTED] at the latest. .... | 1169 |
| 1.  | Claim 1.....   | 1169 |
| 2.  | Claim 2.....   | 1180 |
| 3.  | Claim 3.....   | 1180 |
| 4.  | Claim 4.....   | 1181 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 5.  | Claim 6.....   | 1182 |
| 6.  | Claim 7.....   | 1182 |
| 7.  | Claim 10.....  | 1186 |
| 8.  | Claim 11.....  | 1187 |
| 9.  | Claim 12.....  | 1187 |
| 10. | Claim 17.....  | 1190 |
| 11. | Claim 18.....  | 1191 |
| 12. | Claim 19.....  | 1191 |
| 13. | Claim 21.....  | 1191 |
| 14. | Claim 22.....  | 1192 |
| 15. | Claim 24.....  | 1192 |
| 16. | Claim 26.....  | 1192 |
| 17. | Claim 29.....  | 1193 |
| 18. | Claim 30.....  | 1194 |
| 19. | Claim 31.....  | 1196 |
| 20. | Claim 32.....  | 1197 |
| 21. | Claim 33.....  | 1197 |
| 22. | Claim 34.....  | 1197 |
| 23. | Claim 36.....  | 1198 |
| U.  | Anticipation by and/or Obviousness in View of [REDACTED]<br>[REDACTED] at the latest. .... | 1198 |
| 1.  | Claim 1.....   | 1198 |
| 2.  | Claim 2.....   | 1206 |
| 3.  | Claim 3.....   | 1207 |
| 4.  | Claim 4.....   | 1209 |
| 5.  | Claim 6.....   | 1210 |
| 6.  | Claim 7.....   | 1211 |
| 7.  | Claim 10.....  | 1214 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 8.  | Claim 11.....  | 1214 |
| 9.  | Claim 12.....  | 1215 |
| 10. | Claim 16.....  | 1215 |
| 11. | Claim 17.....  | 1215 |
| 12. | Claim 18.....  | 1216 |
| 13. | Claim 19.....  | 1216 |
| 14. | Claim 21.....  | 1217 |
| 15. | Claim 22.....  | 1217 |
| 16. | Claim 24.....  | 1218 |
| 17. | Claim 26.....  | 1218 |
| 18. | Claim 29.....  | 1218 |
| 19. | Claim 30.....  | 1219 |
| 20. | Claim 31.....  | 1221 |
| 21. | Claim 32.....  | 1221 |
| 22. | Claim 33.....  | 1221 |
| 23. | Claim 34.....  | 1222 |
| 24. | Claim 36.....  | 1222 |
| V.  | Anticipation by and/or Obviousness in View of the IBM xSeries Server<br>Software Rejuvenation Agent, marketed by IBM by 2000 at the latest. .... | 1222 |
| 1.  | Claim 1.....   | 1222 |
| 2.  | Claim 2.....   | 1234 |
| 3.  | Claim 3.....   | 1235 |
| 4.  | Claim 4.....   | 1236 |
| 5.  | Claim 6.....   | 1237 |
| 6.  | Claim 7.....   | 1238 |
| 7.  | Claim 10.....  | 1242 |
| 8.  | Claim 11.....  | 1242 |
| 9.  | Claim 12.....  | 1243 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 10. | Claim 16.....  | 1243 |
| 11. | Claim 17.....  | 1244 |
| 12. | Claim 18.....  | 1244 |
| 13. | Claim 19.....  | 1245 |
| 14. | Claim 21.....  | 1245 |
| 15. | Claim 22.....  | 1245 |
| 16. | Claim 24.....  | 1247 |
| 17. | Claim 26.....  | 1247 |
| 18. | Claim 29.....  | 1247 |
| 19. | Claim 30.....  | 1248 |
| 20. | Claim 31.....  | 1250 |
| 21. | Claim 32.....  | 1251 |
| 22. | Claim 33.....  | 1251 |
| 23. | Claim 34.....  | 1252 |
| 24. | Claim 36.....  | 1252 |
| W.  | Anticipation by and/or Obviousness in View of United States Patent No. 6,826,150, filed 10/2/2001 and issued to Bhattacharya, et al. on 11/30/2004. .... | 1252 |
| 1.  | Claim 1.....   | 1252 |
| 2.  | Claim 2.....   | 1259 |
| 3.  | Claim 3.....   | 1259 |
| 4.  | Claim 4.....   | 1261 |
| 5.  | Claim 6.....   | 1262 |
| 6.  | Claim 7.....   | 1262 |
| 7.  | Claim 10.....  | 1265 |
| 8.  | Claim 11.....  | 1265 |
| 9.  | Claim 12.....  | 1266 |
| 10. | Claim 16.....  | 1267 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 11. | Claim 17.....  | 1267 |
| 12. | Claim 18.....  | 1268 |
| 13. | Claim 19.....  | 1268 |
| 14. | Claim 21.....  | 1269 |
| 15. | Claim 22.....  | 1270 |
| 16. | Claim 24.....  | 1271 |
| 17. | Claim 26.....  | 1271 |
| 18. | Claim 29.....  | 1272 |
| 19. | Claim 30.....  | 1272 |
| 20. | Claim 31.....  | 1275 |
| 21. | Claim 32.....  | 1275 |
| 22. | Claim 33.....  | 1275 |
| 23. | Claim 34.....  | 1276 |
| 24. | Claim 36.....  | 1276 |
| X.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,219,719, filed on 1/22/1997 and issued to Graf on 4/17/2001..... | 1276 |
| 1.  | Claim 1.....   | 1276 |
| 2.  | Claim 2.....   | 1283 |
| 3.  | Claim 3.....   | 1284 |
| 4.  | Claim 4.....   | 1285 |
| 5.  | Claim 6.....   | 1285 |
| 6.  | Claim 7.....   | 1286 |
| 7.  | Claim 10.....  | 1289 |
| 8.  | Claim 11.....  | 1289 |
| 9.  | Claim 12.....  | 1289 |
| 10. | Claim 16.....  | 1290 |
| 11. | Claim 17.....  | 1290 |
| 12. | Claim 18.....  | 1291 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 13. | Claim 19.....  | 1292 |
| 14. | Claim 21.....  | 1292 |
| 15. | Claim 22.....  | 1293 |
| 16. | Claim 24.....  | 1293 |
| 17. | Claim 26.....  | 1293 |
| 18. | Claim 29.....  | 1294 |
| 19. | Claim 30.....  | 1294 |
| 20. | Claim 31.....  | 1296 |
| 21. | Claim 32.....  | 1297 |
| 22. | Claim 33.....  | 1297 |
| 23. | Claim 34.....  | 1298 |
| 24. | Claim 36.....  | 1298 |
| Y.  | Anticipation by and/or Obviousness in View of United States Patent No.<br>6,243,667, filed 5/28/1996 and issued to Kerr, et al. on 6/5/2001..... | 1298 |
| 1.  | Claim 1.....   | 1298 |
| 2.  | Claim 2.....   | 1303 |
| 3.  | Claim 3.....   | 1304 |
| 4.  | Claim 4.....   | 1305 |
| 5.  | Claim 6.....   | 1305 |
| 6.  | Claim 7.....   | 1306 |
| 7.  | Claim 10.....  | 1308 |
| 8.  | Claim 11.....  | 1308 |
| 9.  | Claim 12.....  | 1309 |
| 10. | Claim 16.....  | 1309 |
| 11. | Claim 17.....  | 1309 |
| 12. | Claim 18.....  | 1310 |
| 13. | Claim 19.....  | 1311 |
| 14. | Claim 21.....  | 1312 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 15. | Claim 22.....  | 1312 |
| 16. | Claim 24.....  | 1313 |
| 17. | Claim 26.....  | 1313 |
| 18. | Claim 29.....  | 1313 |
| 19. | Claim 30.....  | 1314 |
| 20. | Claim 31.....  | 1316 |
| 21. | Claim 32.....  | 1316 |
| 22. | Claim 33.....  | 1316 |
| 23. | Claim 34.....  | 1317 |
| 24. | Claim 36.....  | 1317 |
| Z.  | Anticipation by and/or Obviousness in View of Cisco NetFlow, marketed by Cisco by 1995 at the latest. .... | 1317 |
| 1.  | Claim 1.....   | 1318 |
| 2.  | Claim 2.....   | 1322 |
| 3.  | Claim 3.....   | 1323 |
| 4.  | Claim 4.....   | 1324 |
| 5.  | Claim 6.....   | 1324 |
| 6.  | Claim 7.....   | 1325 |
| 7.  | Claim 10.....  | 1327 |
| 8.  | Claim 11.....  | 1327 |
| 9.  | Claim 12.....  | 1328 |
| 10. | Claim 16.....  | 1328 |
| 11. | Claim 17.....  | 1328 |
| 12. | Claim 18.....  | 1329 |
| 13. | Claim 19.....  | 1330 |
| 14. | Claim 21.....  | 1330 |
| 15. | Claim 22.....  | 1331 |
| 16. | Claim 24.....  | 1331 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|            |  |             |
|------------|--|-------------|
| 17.        | Claim 26.....  | 1331        |
| 18.        | Claim 29.....  | 1332        |
| 19.        | Claim 30.....  | 1332        |
| 20.        | Claim 31.....  | 1334        |
| 21.        | Claim 32.....  | 1335        |
| 22.        | Claim 33.....  | 1335        |
| 23.        | Claim 34.....  | 1335        |
| 24.        | Claim 36.....  | 1336        |
| <b>XI.</b> | <b>PRIOR ART COMBINATIONS RENDERING THE '730 PATENT<br/>OBVIOUS .....</b>              | <b>1336</b> |
| A.         | NetRanger in Combination with Goldman .....  | 1336        |
| B.         | Cisco Catalyst 5000 Switch Family in combination with NetRanger and/or<br>Goldman..... | 1337        |
| C.         | Goldman in combination with Kasteleijn and/or Natarajan and/or RFC<br>2328.....        | 1337        |
| 1.         | Motivation to Combine Goldman with Kasteleijn.....                                     | 1337        |
| 2.         | Motivation to Combine Goldman with Natarajan .....                                     | 1337        |
| 3.         | Motivation to Combine Goldman with RFC 2328 .....                                      | 1337        |
| 4.         | Claim 1 .....  | 1338        |
| 5.         | Claim 2.....   | 1340        |
| 6.         | Claim 3.....   | 1340        |
| 7.         | Claim 4.....   | 1341        |
| 8.         | Claim 6.....   | 1341        |
| 9.         | Claim 7.....   | 1343        |
| 10.        | Claim 10.....  | 1345        |
| 11.        | Claim 11.....  | 1345        |
| 12.        | Claim 12.....  | 1346        |
| 13.        | Claim 13.....  | 1346        |
| 14.        | Claim 16.....  | 1346        |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 15. | Claim 17.....  | 1347 |
| 16. | Claim 18.....  | 1347 |
| 17. | Claim 19.....  | 1347 |
| 18. | Claim 21.....  | 1347 |
| 19. | Claim 22.....  | 1348 |
| 20. | Claim 24.....  | 1348 |
| 21. | Claim 26.....  | 1348 |
| 22. | Claim 29.....  | 1348 |
| 23. | Claim 30.....  | 1349 |
| 24. | Claim 31.....  | 1351 |
| 25. | Claim 32.....  | 1351 |
| 26. | Claim 33.....  | 1351 |
| 27. | Claim 34.....  | 1352 |
| 28. | Claim 36.....  | 1352 |
| D.  | Pandya in combination with Kasteleijn and/or Goldman and/or Douik<br>and/or RFC 2328/Afek..... | 1352 |
| 1.  | Motivation to Combine Pandya with Kasteleijn.....  | 1352 |
| 2.  | Motivation to Combine Pandya with Goldman .....  | 1352 |
| 3.  | Motivation to Combine Pandya with Douik .....  | 1353 |
| 4.  | Motivation to Combine Pandya with RFC 2328 and/or Afek .....                                   | 1353 |
| 5.  | Claim 1 .....  | 1354 |
| 6.  | Claim 2.....   | 1356 |
| 7.  | Claim 3.....   | 1356 |
| 8.  | Claim 4.....   | 1356 |
| 9.  | Claim 6.....   | 1357 |
| 10. | Claim 7.....   | 1359 |
| 11. | Claim 10.....  | 1361 |
| 12. | Claim 11.....  | 1361 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 13. | Claim 12.....   | 1361 |
| 14. | Claim 13.....   | 1362 |
| 15. | Claim 16.....   | 1362 |
| 16. | Claim 17.....   | 1362 |
| 17. | Claim 18.....   | 1362 |
| 18. | Claim 19.....   | 1363 |
| 19. | Claim 21.....   | 1363 |
| 20. | Claim 22.....   | 1363 |
| 21. | Claim 24.....   | 1363 |
| 22. | Claim 26.....   | 1364 |
| 23. | Claim 29.....   | 1364 |
| 24. | Claim 30.....   | 1364 |
| 25. | Claim 31.....   | 1366 |
| 26. | Claim 32.....   | 1366 |
| 27. | Claim 33.....   | 1367 |
| 28. | Claim 34.....   | 1367 |
| 29. | Claim 36.....   | 1367 |
| E.  | Douik in combination with Kasteleijn and/or Turek and/or Afek and/or Goldman..... | 1368 |
| 1.  | Motivation to Combine Douik with Kasteleijn.....                                  | 1368 |
| 2.  | Motivation to Combine Douik with Turek .....                                      | 1368 |
| 3.  | Motivation to Combine Douik with Afek.....  | 1369 |
| 4.  | Motivation to Combine Douik with Goldman.....                                     | 1369 |
| 5.  | Claim 1.....  | 1369 |
| 6.  | Claim 2.....  | 1370 |
| 7.  | Claim 3.....  | 1370 |
| 8.  | Claim 4.....  | 1371 |
| 9.  | Claim 6.....  | 1372 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 10. | Claim 7.....   | 1373 |
| 11. | Claim 10.....  | 1376 |
| 12. | Claim 11.....  | 1376 |
| 13. | Claim 12.....  | 1376 |
| 14. | Claim 13.....  | 1376 |
| 15. | Claim 16.....  | 1377 |
| 16. | Claim 17.....  | 1377 |
| 17. | Claim 18.....  | 1377 |
| 18. | Claim 19.....  | 1377 |
| 19. | Claim 21.....  | 1378 |
| 20. | Claim 22.....  | 1378 |
| 21. | Claim 24.....  | 1378 |
| 22. | Claim 26.....  | 1378 |
| 23. | Claim 29.....  | 1378 |
| 24. | Claim 30.....  | 1379 |
| 25. | Claim 31.....  | 1381 |
| 26. | Claim 32.....  | 1381 |
| 27. | Claim 33.....  | 1381 |
| 28. | Claim 34.....  | 1382 |
| 29. | Claim 36.....  | 1382 |
| F.  | Natarajan in combination with Kasteleijn and/or Goldman and/or Turek<br>and/or Yoshihara and/or RFC 2328 ..... | 1382 |
| 1.  | Motivation to Combine Natarajan with Kasteleijn .....  | 1382 |
| 2.  | Motivation to Combine Natarajan with Goldman .....   | 1383 |
| 3.  | Motivation to Combine Natarajan with Turek.....  | 1383 |
| 4.  | Motivation to Combine Natarajan with Yoshihara.....  | 1383 |
| 5.  | Motivation to Combine Natarajan with RFC 2328.....   | 1383 |
| 6.  | Claim 1 .....  | 1384 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 7.  | Claim 2.....  | 1385 |
| 8.  | Claim 3.....  | 1386 |
| 9.  | Claim 4.....  | 1386 |
| 10. | Claim 6.....  | 1387 |
| 11. | Claim 7.....  | 1387 |
| 12. | Claim 10.....   | 1389 |
| 13. | Claim 11.....   | 1389 |
| 14. | Claim 12.....   | 1390 |
| 15. | Claim 13.....   | 1390 |
| 16. | Claim 16.....   | 1390 |
| 17. | Claim 17.....   | 1391 |
| 18. | Claim 18.....   | 1391 |
| 19. | Claim 19.....   | 1391 |
| 20. | Claim 21.....   | 1391 |
| 21. | Claim 22.....   | 1392 |
| 22. | Claim 24.....   | 1392 |
| 23. | Claim 26.....   | 1392 |
| 24. | Claim 29.....   | 1392 |
| 25. | Claim 30.....   | 1393 |
| 26. | Claim 31.....   | 1395 |
| 27. | Claim 32.....   | 1395 |
| 28. | Claim 33.....   | 1396 |
| 29. | Claim 34.....   | 1396 |
| 30. | Claim 36.....   | 1396 |
| G.  | Bonnell in combination with Kasteleijn and/or Goldman and/or Hellerstein<br>and/or Turk and/or RFC 2328/Afek..... | 1396 |
| 1.  | Motivation to Combine Bonnell with Kasteleijn.....  | 1397 |
| 2.  | Motivation to Combine Bonnell with Goldman .....  | 1397 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 3.  | Motivation to Combine Bonnell with Hellerstein.....                                    | 1397 |
| 4.  | Motivation to Combine Bonnell with Turek.....  | 1398 |
| 5.  | Motivation to Combine Motivation to Combine Bonnell with RFC<br>2328 and/or Afek ..... | 1398 |
| 6.  | Claim 1.....   | 1398 |
| 7.  | Claim 2.....   | 1400 |
| 8.  | Claim 3.....   | 1401 |
| 9.  | Claim 4.....   | 1401 |
| 10. | Claim 6.....   | 1402 |
| 11. | Claim 7.....   | 1404 |
| 12. | Claim 10.....  | 1406 |
| 13. | Claim 11.....  | 1406 |
| 14. | Claim 12.....  | 1406 |
| 15. | Claim 13.....  | 1407 |
| 16. | Claim 16.....  | 1407 |
| 17. | Claim 17.....  | 1407 |
| 18. | Claim 18.....  | 1407 |
| 19. | Claim 19.....  | 1408 |
| 20. | Claim 21.....  | 1408 |
| 21. | Claim 22.....  | 1408 |
| 22. | Claim 24.....  | 1408 |
| 23. | Claim 26.....  | 1408 |
| 24. | Claim 29.....  | 1409 |
| 25. | Claim 30.....  | 1409 |
| 26. | Claim 31.....  | 1411 |
| 27. | Claim 32.....  | 1411 |
| 28. | Claim 33.....  | 1412 |
| 29. | Claim 34.....  | 1412 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 30. | Claim 36.....  | 1412 |
| H.  | Boukobza in combination with Goldman and/or Hellerstein and/or Kasteleijn and/or RFC 2328/Afek ..... | 1412 |
| 1.  | Motivation to Combine Boukobza with Goldman.....   | 1413 |
| 2.  | Motivation to Combine Boukobza with Hellerstein .....  | 1413 |
| 3.  | Motivation to Combine Boukobza with Kasteleijn .....   | 1413 |
| 4.  | Motivation to Combine Boukobza with RFC 2328 and/or Afek.....  | 1413 |
| 5.  | Claim 1 .....  | 1414 |
| 6.  | Claim 2.....   | 1416 |
| 7.  | Claim 3.....   | 1416 |
| 8.  | Claim 4.....   | 1417 |
| 9.  | Claim 6.....   | 1418 |
| 10. | Claim 7.....   | 1420 |
| 11. | Claim 10.....  | 1422 |
| 12. | Claim 11.....  | 1422 |
| 13. | Claim 12.....  | 1423 |
| 14. | Claim 13.....  | 1423 |
| 15. | Claim 16.....  | 1423 |
| 16. | Claim 17.....  | 1423 |
| 17. | Claim 18.....  | 1424 |
| 18. | Claim 19.....  | 1424 |
| 19. | Claim 21.....  | 1424 |
| 20. | Claim 22.....  | 1424 |
| 21. | Claim 24.....  | 1425 |
| 22. | Claim 26.....  | 1425 |
| 23. | Claim 29.....  | 1425 |
| 24. | Claim 30.....  | 1425 |
| 25. | Claim 31.....  | 1427 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 26. | Claim 32.....  | 1428 |
| 27. | Claim 33.....  | 1428 |
| 28. | Claim 34.....  | 1429 |
| 29. | Claim 36.....  | 1429 |
| I.  | Hellerstein in combination with Turek and/or RFC 2328/Afek and/or<br>Goldman and/or Kasteleijn ..... | 1429 |
| 1.  | Motivation to Combine Hellerstein with Turek.....  | 1429 |
| 2.  | Motivation to Combine Hellerstein with RFC 2328 and/or Afek.....                                     | 1429 |
| 3.  | Motivation to Combine Hellerstein with Goldman .....   | 1430 |
| 4.  | Motivation to Combine Hellerstein with Kasteleijn .....  | 1430 |
| 5.  | Claim 1 .....  | 1430 |
| 6.  | Claim 2.....   | 1432 |
| 7.  | Claim 3.....   | 1432 |
| 8.  | Claim 4.....   | 1433 |
| 9.  | Claim 6.....   | 1433 |
| 10. | Claim 7.....   | 1435 |
| 11. | Claim 10.....  | 1437 |
| 12. | Claim 11.....  | 1437 |
| 13. | Claim 12.....  | 1438 |
| 14. | Claim 13.....  | 1438 |
| 15. | Claim 16.....  | 1438 |
| 16. | Claim 17.....  | 1438 |
| 17. | Claim 18.....  | 1439 |
| 18. | Claim 19.....  | 1439 |
| 19. | Claim 21.....  | 1439 |
| 20. | Claim 22.....  | 1440 |
| 21. | Claim 24.....  | 1440 |
| 22. | Claim 26.....  | 1440 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 23. | Claim 29.....   | 1440 |
| 24. | Claim 30.....   | 1441 |
| 25. | Claim 31.....   | 1443 |
| 26. | Claim 32.....   | 1443 |
| 27. | Claim 33.....   | 1443 |
| 28. | Claim 34.....   | 1444 |
| 29. | Claim 36.....   | 1444 |
| J.  | Kasteleijn in combination with Douik and/or Douik and RFC 2328 and/or<br>Goldman and/or Hellerstein and/or Turek and/or RFC 2328/Afek ..... | 1444 |
| 1.  | Motivation to Combine Kasteleijn with Douik.....  | 1444 |
| 2.  | Motivation to Combine Kasteleijn with Douik and RFC 2328 .....  | 1445 |
| 3.  | Motivation to Combine Kasteleijn with Goldman.....  | 1445 |
| 4.  | Motivation to Combine Kasteleijn with Hellerstein .....   | 1445 |
| 5.  | Motivation to Combine Kasteleijn with Turek .....   | 1446 |
| 6.  | Motivation to Combine Kasteleijn with RFC 2328 and/or Afek.....   | 1446 |
| 7.  | Claim 1 .....   | 1446 |
| 8.  | Claim 2.....  | 1448 |
| 9.  | Claim 3.....  | 1448 |
| 10. | Claim 4.....  | 1449 |
| 11. | Claim 6.....  | 1450 |
| 12. | Claim 7.....  | 1451 |
| 13. | Claim 10.....   | 1453 |
| 14. | Claim 11.....   | 1454 |
| 15. | Claim 12.....   | 1454 |
| 16. | Claim 13.....   | 1454 |
| 17. | Claim 16.....   | 1454 |
| 18. | Claim 17.....   | 1455 |
| 19. | Claim 18.....   | 1455 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 20. | Claim 19.....  | 1455 |
| 21. | Claim 21.....  | 1455 |
| 22. | Claim 22.....  | 1456 |
| 23. | Claim 24.....  | 1456 |
| 24. | Claim 26.....  | 1456 |
| 25. | Claim 29.....  | 1456 |
| 26. | Claim 30.....  | 1457 |
| 27. | Claim 31.....  | 1459 |
| 28. | Claim 32.....  | 1459 |
| 29. | Claim 33.....  | 1459 |
| 30. | Claim 34.....  | 1460 |
| 31. | Claim 36.....  | 1460 |
| K.  | da Rocha in combination with Kasteleijn and/or Goldman and/or Turek<br>and/or RFC 2328/Afek..... | 1460 |
| 1.  | Motivation to Combine Da Rocha with Kasteleijn.....  | 1460 |
| 2.  | Motivation to Combine Da Rocha with Goldman .....  | 1461 |
| 3.  | Motivation to Combine Da Rocha with Turek .....  | 1461 |
| 4.  | Motivation to Combine Da Rocha with RFC 2328 and/or Afek .....                                   | 1461 |
| 5.  | Claim 1.....   | 1462 |
| 6.  | Claim 2.....   | 1464 |
| 7.  | Claim 3.....   | 1464 |
| 8.  | Claim 4.....   | 1465 |
| 9.  | Claim 6.....   | 1465 |
| 10. | Claim 7.....   | 1467 |
| 11. | Claim 10.....  | 1469 |
| 12. | Claim 11.....  | 1469 |
| 13. | Claim 12.....  | 1470 |
| 14. | Claim 13.....  | 1470 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 15. | Claim 16.....   | 1470 |
| 16. | Claim 17.....   | 1470 |
| 17. | Claim 18.....   | 1471 |
| 18. | Claim 19.....   | 1471 |
| 19. | Claim 21.....   | 1471 |
| 20. | Claim 22.....   | 1472 |
| 21. | Claim 24.....   | 1472 |
| 22. | Claim 26.....   | 1472 |
| 23. | Claim 29.....   | 1472 |
| 24. | Claim 30.....   | 1473 |
| 25. | Claim 31.....   | 1475 |
| 26. | Claim 32.....   | 1475 |
| 27. | Claim 33.....   | 1475 |
| 28. | Claim 34.....   | 1476 |
| 29. | Claim 36.....   | 1476 |
| L.  | INCA in combination with Kasteleijn and/or Goldman and/or Hellerstein<br>and/or RFC 2328/Afek and/or Turek..... | 1476 |
| 1.  | Motivation to Combine INCA with Kasteleijn.....   | 1476 |
| 2.  | Motivation to Combine INCA with Goldman .....   | 1477 |
| 3.  | Motivation to Combine INCA with Hellerstein.....  | 1477 |
| 4.  | Motivation to Combine INCA with RFC 2328 and/or Afek .....  | 1477 |
| 5.  | Motivation to Combine INCA with Turek.....  | 1478 |
| 6.  | Claim 1.....  | 1478 |
| 7.  | Claim 2.....  | 1480 |
| 8.  | Claim 3.....  | 1480 |
| 9.  | Claim 4.....  | 1481 |
| 10. | Claim 6.....  | 1481 |
| 11. | Claim 7.....  | 1483 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 12. | Claim 10.....  | 1485 |
| 13. | Claim 11.....  | 1485 |
| 14. | Claim 12.....  | 1486 |
| 15. | Claim 13.....  | 1486 |
| 16. | Claim 16.....  | 1486 |
| 17. | Claim 17.....  | 1486 |
| 18. | Claim 18.....  | 1487 |
| 19. | Claim 19.....  | 1487 |
| 20. | Claim 21.....  | 1487 |
| 21. | Claim 22.....  | 1487 |
| 22. | Claim 24.....  | 1488 |
| 23. | Claim 26.....  | 1488 |
| 24. | Claim 29.....  | 1488 |
| 25. | Claim 30.....  | 1488 |
| 26. | Claim 31.....  | 1490 |
| 27. | Claim 32.....  | 1491 |
| 28. | Claim 33.....  | 1491 |
| 29. | Claim 34.....  | 1492 |
| 30. | Claim 36.....  | 1492 |
| M.  | Boudaoud in combination with Kasteleijn and/or RFC 2328/Afek and/or Goldman..... | 1492 |
| 1.  | Motivation to Combine Boudaoud with Kasteleijn .....                             | 1492 |
| 2.  | Motivation to Combine Boudaoud with RFC 2328 and/or Afek.....                    | 1492 |
| 3.  | Motivation to Combine Boudaoud with Goldman.....                                 | 1493 |
| 4.  | Claim 1 .....  | 1493 |
| 5.  | Claim 2.....   | 1494 |
| 6.  | Claim 3.....   | 1495 |
| 7.  | Claim 4.....   | 1495 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 8.  | Claim 6.....  | 1496 |
| 9.  | Claim 7.....  | 1498 |
| 10. | Claim 10.....   | 1500 |
| 11. | Claim 11.....   | 1500 |
| 12. | Claim 12.....   | 1500 |
| 13. | Claim 13.....   | 1500 |
| 14. | Claim 16.....   | 1501 |
| 15. | Claim 17.....   | 1501 |
| 16. | Claim 18.....   | 1501 |
| 17. | Claim 19.....   | 1501 |
| 18. | Claim 21.....   | 1502 |
| 19. | Claim 22.....   | 1502 |
| 20. | Claim 24.....   | 1502 |
| 21. | Claim 26.....   | 1502 |
| 22. | Claim 29.....   | 1503 |
| 23. | Claim 30.....   | 1503 |
| 24. | Claim 31.....   | 1505 |
| 25. | Claim 32.....   | 1505 |
| 26. | Claim 33.....   | 1506 |
| 27. | Claim 34.....   | 1506 |
| 28. | Claim 36.....   | 1506 |
| N.  | Yoshihara in combination with Kasteleijn and/or Goldman and/or Castelli<br>and/or Turek and/or Natarajan and/or RFC 2328/Afek ..... | 1507 |
| 1.  | Motivation to Combine Yoshihara with Kasteleijn .....   | 1507 |
| 2.  | Motivation to Combine Yoshihara with Goldman.....   | 1507 |
| 3.  | Motivation to Combine Yoshihara with Castelli .....   | 1507 |
| 4.  | Motivation to Combine Yoshihara with Turek.....   | 1507 |
| 5.  | Motivation to Combine Yoshihara with Natarajan.....   | 1508 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 6.  | Motivation to Combine Yoshihara with RFC 2328 and/or Afek.....  | 1508 |
| 7.  | Claim 1.....  | 1508 |
| 8.  | Claim 2.....  | 1510 |
| 9.  | Claim 3.....  | 1510 |
| 10. | Claim 4.....  | 1511 |
| 11. | Claim 6.....  | 1512 |
| 12. | Claim 7.....  | 1513 |
| 13. | Claim 10.....   | 1515 |
| 14. | Claim 11.....   | 1516 |
| 15. | Claim 12.....   | 1516 |
| 16. | Claim 13.....   | 1516 |
| 17. | Claim 16.....   | 1516 |
| 18. | Claim 17.....   | 1517 |
| 19. | Claim 18.....   | 1517 |
| 20. | Claim 19.....   | 1517 |
| 21. | Claim 21.....   | 1518 |
| 22. | Claim 22.....   | 1518 |
| 23. | Claim 24.....   | 1518 |
| 24. | Claim 26.....   | 1518 |
| 25. | Claim 29.....   | 1518 |
| 26. | Claim 30.....   | 1519 |
| 27. | Claim 31.....   | 1521 |
| 28. | Claim 32.....   | 1521 |
| 29. | Claim 33.....   | 1521 |
| 30. | Claim 34.....   | 1522 |
| 31. | Claim 36.....   | 1522 |
| O.  | Cisco NetRanger in combination with Kasteleijn, and/or Goldman, and/or de Rocha, and/or RFC 2328..... | 1522 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 1.  | Motivation to Combine Cisco NetRanger with Kasteleijn ..... | 1522 |
| 2.  | Motivation to combine Cisco NetRanger with Goldman .....    | 1523 |
| 3.  | Motivation to combine Cisco NetRanger with de Rocha .....   | 1523 |
| 4.  | Motivation to combine Cisco NetRanger with RFC 2328.....    | 1523 |
| 5.  | Claim 1 .....   | 1524 |
| 6.  | Claim 2.....  | 1525 |
| 7.  | Claim 3.....  | 1526 |
| 8.  | Claim 4.....  | 1526 |
| 9.  | Claim 6.....  | 1527 |
| 10. | Claim 7.....  | 1528 |
| 11. | Claim 10.....   | 1530 |
| 12. | Claim 11.....   | 1530 |
| 13. | Claim 12.....   | 1530 |
| 14. | Claim 13.....   | 1530 |
| 15. | Claim 16.....   | 1531 |
| 16. | Claim 17.....   | 1531 |
| 17. | Claim 18.....   | 1531 |
| 18. | Claim 19.....   | 1531 |
| 19. | Claim 21.....   | 1532 |
| 20. | Claim 22.....   | 1532 |
| 21. | Claim 24.....   | 1532 |
| 22. | Claim 26.....   | 1532 |
| 23. | Claim 29.....   | 1533 |
| 24. | Claim 30.....   | 1533 |
| 25. | Claim 31.....   | 1535 |
| 26. | Claim 32.....   | 1535 |
| 27. | Claim 33.....   | 1535 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 28. | Claim 34.....   | 1536 |
| 29. | Claim 36.....   | 1536 |
| P.  | Cisco Catalyst in combination with Kasteleijn, and/or Goldman, and/or de Rocha, and/or NetRanger..... | 1536 |
| 1.  | Motivation to Combine Cisco Catalyst with Kasteleijn .....  | 1537 |
| 2.  | Motivation to combine Cisco Catalyst with Goldman.....  | 1537 |
| 3.  | Motivation to combine Cisco Catalyst with de Rocha.....   | 1537 |
| 4.  | Motivation to combine Cisco Catalyst with NetRanger .....   | 1538 |
| 5.  | Claim 1 .....   | 1538 |
| 6.  | Claim 2.....  | 1540 |
| 7.  | Claim 3.....  | 1540 |
| 8.  | Claim 4.....  | 1541 |
| 9.  | Claim 6.....  | 1541 |
| 10. | Claim 7.....  | 1541 |
| 11. | Claim 10.....   | 1543 |
| 12. | Claim 11.....   | 1544 |
| 13. | Claim 12.....   | 1544 |
| 14. | Claim 13.....   | 1544 |
| 15. | Claim 16.....   | 1544 |
| 16. | Claim 17.....   | 1545 |
| 17. | Claim 18.....   | 1545 |
| 18. | Claim 19.....   | 1545 |
| 19. | Claim 21.....   | 1546 |
| 20. | Claim 22.....   | 1546 |
| 21. | Claim 24.....   | 1546 |
| 22. | Claim 26.....   | 1546 |
| 23. | Claim 29.....   | 1547 |
| 24. | Claim 30.....   | 1547 |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 25. | Claim 31.....   | 1549 |
| 26. | Claim 32.....   | 1549 |
| 27. | Claim 33.....   | 1550 |
| 28. | Claim 34.....   | 1550 |
| 29. | Claim 36.....   | 1550 |
| Q.  | ██████████ in combination with Kasteleijn, and/or Pandya, and/or de Rocha, and/or Hellerstein, and/or RFC 2328..... | 1551 |
| 1.  | Motivation to Combine ██████████ with Kasteleijn .....  | 1551 |
| 2.  | Motivation to combine ██████████ with Pandya.....   | 1551 |
| 3.  | Motivation to combine ██████████ with de Rocha.....   | 1551 |
| 4.  | Motivation to combine ██████████ with NetRanger .....   | 1552 |
| 5.  | Motivation to combine ██████████ with RFC 2328 .....  | 1552 |
| 6.  | Claim 1 .....   | 1553 |
| 7.  | Claim 2.....  | 1555 |
| 8.  | Claim 3.....  | 1555 |
| 9.  | Claim 4.....  | 1556 |
| 10. | Claim 6.....  | 1556 |
| 11. | Claim 7.....  | 1558 |
| 12. | Claim 10.....   | 1560 |
| 13. | Claim 11.....   | 1560 |
| 14. | Claim 12.....   | 1560 |
| 15. | Claim 13.....   | 1561 |
| 16. | Claim 16.....   | 1561 |
| 17. | Claim 17.....   | 1561 |
| 18. | Claim 18.....   | 1562 |
| 19. | Claim 19.....   | 1562 |
| 20. | Claim 21.....   | 1562 |
| 21. | Claim 22.....   | 1563 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 22. | Claim 24.....  | 1563 |
| 23. | Claim 26.....  | 1563 |
| 24. | Claim 29.....  | 1563 |
| 25. | Claim 30.....  | 1564 |
| 26. | Claim 31.....  | 1566 |
| 27. | Claim 32.....  | 1566 |
| 28. | Claim 33.....  | 1566 |
| 29. | Claim 34.....  | 1567 |
| 30. | Claim 36.....  | 1567 |
| R.  | Cisco Receive rACL in combination with Kasteleijn and/or NetRanger<br>and/or Goldman .....   | 1567 |
| 1.  | Motivation to Combine Cisco Receive rACL in combination with<br>Kasteleijn .....   | 1567 |
| 2.  | Motivation to Combine Cisco Receive rACL in combination with<br>NetRanger .....  | 1568 |
| 3.  | Motivation to Combine Cisco Receive rACL in combination with<br>Goldman.....   | 1568 |
| 4.  | For motivation to combine with RFC 2328 and/or Afek: copy and<br>paste the following; delete reference to RFC or Afek if only one of<br>the two references is covered in the invalidity chart..... | 1568 |
| 5.  | Claim 1 .....  | 1569 |
| 6.  | Claim 2.....   | 1572 |
| 7.  | Claim 3.....   | 1572 |
| 8.  | Claim 4.....   | 1573 |
| 9.  | Claim 6.....   | 1573 |
| 10. | Claim 7.....   | 1573 |
| 11. | Claim 10.....  | 1575 |
| 12. | Claim 11.....  | 1576 |
| 13. | Claim 12.....  | 1576 |
| 14. | Claim 16.....  | 1576 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 15. | Claim 17.....   | 1576 |
| 16. | Claim 18.....   | 1577 |
| 17. | Claim 19.....   | 1577 |
| 18. | Claim 21.....   | 1577 |
| 19. | Claim 22.....   | 1577 |
| 20. | Claim 24.....   | 1578 |
| 21. | Claim 26.....   | 1578 |
| 22. | Claim 29.....   | 1578 |
| 23. | Claim 30.....   | 1579 |
| 24. | Claim 31.....   | 1581 |
| 25. | Claim 32.....   | 1581 |
| 26. | Claim 33.....   | 1581 |
| 27. | Claim 34.....   | 1582 |
| 28. | Claim 36.....   | 1582 |
| S.  | NetRanger in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek .....                        | 1582 |
| 1.  | Motivation to Combine Cisco NetRanger with Buchanan, Cantrill, and/or Berry .....                       | 1582 |
| 2.  | Motivation to Combine Cisco NetRanger with RFC 2328 and/or Afek.....                                    | 1583 |
| T.  | Conti in combination with Kasteleijn and/or Goldman and/or Hellerstein and/or RFC 2328 and/or Afek..... | 1584 |
| 1.  | Motivation to Combine Conti with Kasteleijn.....  | 1584 |
| 2.  | Motivation to Combine Conti with Goldman .....  | 1585 |
| 3.  | Motivation to Combine Conti with Hellerstein .....  | 1585 |
| 4.  | Motivation to Combine Conti with RFC 2328 and/or Afek .....   | 1585 |
| 5.  | Claim 1.....  | 1586 |
| 6.  | Claim 2.....  | 1588 |
| 7.  | Claim 3.....  | 1588 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 8.  | Claim 4.....  | 1589 |
| 9.  | Claim 6.....  | 1589 |
| 10. | Claim 7.....  | 1591 |
| 11. | Claim 10.....   | 1593 |
| 12. | Claim 11.....   | 1593 |
| 13. | Claim 12.....   | 1593 |
| 14. | Claim 13.....   | 1594 |
| 15. | Claim 16.....   | 1594 |
| 16. | Claim 17.....   | 1594 |
| 17. | Claim 18.....   | 1594 |
| 18. | Claim 19.....   | 1595 |
| 19. | Claim 21.....   | 1595 |
| 20. | Claim 22.....   | 1595 |
| 21. | Claim 24.....   | 1595 |
| 22. | Claim 26.....   | 1596 |
| 23. | Claim 29.....   | 1596 |
| 24. | Claim 30.....   | 1596 |
| 25. | Claim 31.....   | 1598 |
| 26. | Claim 32.....   | 1598 |
| 27. | Claim 33.....   | 1599 |
| 28. | Claim 34.....   | 1599 |
| 29. | Claim 36.....   | 1599 |
| U.  | Castelli in combination with Kasteleijn and/or Turek and/or Goldman<br>and/or RFC 2328 and/or Afek..... | 1600 |
| 1.  | Motivation to Combine Castelli with Kasteleijn .....  | 1600 |
| 2.  | Motivation to Combine Castelli with Turek .....   | 1600 |
| 3.  | Motivation to Combine Castelli with Goldman.....  | 1600 |
| 4.  | Motivation to Combine Castelli with RFC 2328 and/or Afek.....   | 1601 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 5.  | Claim 1.....  | 1601 |
| 6.  | Claim 2.....  | 1603 |
| 7.  | Claim 3.....  | 1603 |
| 8.  | Claim 4.....  | 1604 |
| 9.  | Claim 6.....  | 1604 |
| 10. | Claim 7.....  | 1606 |
| 11. | Claim 10.....                                       | 1608 |
| 12. | Claim 11.....                                       | 1608 |
| 13. | Claim 12.....                                       | 1608 |
| 14. | Claim 13.....                                       | 1609 |
| 15. | Claim 16.....                                       | 1609 |
| 16. | Claim 17.....                                       | 1609 |
| 17. | Claim 18.....                                       | 1609 |
| 18. | Claim 19.....                                       | 1610 |
| 19. | Claim 21.....                                       | 1610 |
| 20. | Claim 22.....                                       | 1610 |
| 21. | Claim 24.....                                       | 1610 |
| 22. | Claim 26.....                                       | 1611 |
| 23. | Claim 29.....                                       | 1611 |
| 24. | Claim 30.....                                       | 1611 |
| 25. | Claim 31.....                                       | 1613 |
| 26. | Claim 32.....                                       | 1613 |
| 27. | Claim 33.....                                       | 1614 |
| 28. | Claim 34.....                                       | 1614 |
| 29. | Claim 36.....                                       | 1614 |
| V.  | Lindskog in combination with .....                  | 1615 |
| 1.  | Motivation to Combine Lindskog with Kasteleijn..... | 1615 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |  |      |
|-----|--|------|
| 2.  | Motivation to Combine Linskog with Goldman.....              | 1615 |
| 3.  | Motivation to Combine Linskog with Hellerstein .....         | 1616 |
| 4.  | Motivation to Combine Linskog with Turek .....               | 1616 |
| 5.  | Motivation to Combine Linskog with RFC 2328 and/or Afek..... | 1616 |
| 6.  | Claim 1.....   | 1617 |
| 7.  | Claim 2.....   | 1619 |
| 8.  | Claim 3.....   | 1619 |
| 9.  | Claim 4.....   | 1620 |
| 10. | Claim 6.....   | 1620 |
| 11. | Claim 7.....   | 1622 |
| 12. | Claim 10.....  | 1624 |
| 13. | Claim 11.....  | 1624 |
| 14. | Claim 12.....  | 1625 |
| 15. | Claim 13.....  | 1625 |
| 16. | Claim 16.....  | 1625 |
| 17. | Claim 17.....  | 1625 |
| 18. | Claim 18.....  | 1626 |
| 19. | Claim 19.....  | 1626 |
| 20. | Claim 21.....  | 1626 |
| 21. | Claim 22.....  | 1627 |
| 22. | Claim 24.....  | 1627 |
| 23. | Claim 26.....  | 1627 |
| 24. | Claim 29.....  | 1627 |
| 25. | Claim 30.....  | 1628 |
| 26. | Claim 31.....  | 1630 |
| 27. | Claim 32.....  | 1630 |
| 28. | Claim 33.....  | 1630 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|     |   |      |
|-----|---|------|
| 29. | Claim 34.....   | 1631 |
| 30. | Claim 36.....   | 1631 |
| W.  | NSMIA in combination with Kasteleijn and/or Hellerstein and/or Goldman and/or RFC 2328 and/or Afek..... | 1631 |
| 1.  | Motivation to Combine NSMIA in combination with Kasteleijn.....   | 1631 |
| 2.  | Motivation to Combine NSMIA in combination with Hellerstein .....                                       | 1632 |
| 3.  | Motivation to Combine NSMIA in combination with Goldman .....   | 1632 |
| 4.  | Claim 1 .....   | 1633 |
| 5.  | Claim 2.....  | 1636 |
| 6.  | Claim 3.....  | 1637 |
| 7.  | Claim 4.....  | 1637 |
| 8.  | Claim 6.....  | 1638 |
| 9.  | Claim 7.....  | 1639 |
| 10. | Claim 10.....   | 1641 |
| 11. | Claim 11.....   | 1642 |
| 12. | Claim 12.....   | 1642 |
| 13. | Claim 16.....   | 1642 |
| 14. | Claim 17.....   | 1642 |
| 15. | Claim 18.....   | 1643 |
| 16. | Claim 19.....   | 1643 |
| 17. | Claim 21.....   | 1643 |
| 18. | Claim 22.....   | 1643 |
| 19. | Claim 24.....   | 1644 |
| 20. | Claim 26.....   | 1644 |
| 21. | Claim 29.....   | 1644 |
| 22. | Claim 30.....   | 1644 |
| 23. | Claim 31.....   | 1647 |
| 24. | Claim 32.....   | 1647 |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

|              |   |             |
|--------------|---|-------------|
| 25.          | Claim 33.....                                     | 1647        |
| 26.          | Claim 34.....                                     | 1648        |
| 27.          | Claim 36.....                                     | 1648        |
| <b>XII.</b>  | <b>INVALIDITY UNDER OTHER GROUNDS .....</b>       | <b>1648</b> |
| A.           | Subject Matter Eligibility - 35 U.S.C. § 101..... | 1648        |
| B.           | Inventorship - 35 U.S.C. § 102(f).....            | 1650        |
| C.           | Invalidity under 35 U.S.C. § 112 .....            | 1653        |
| 1.           | '730 Patent .....                                 | 1653        |
| 2.           | '659 Patent .....                                 | 1665        |
| D.           | Indefiniteness .....                              | 1674        |
| 2.           | '659 Patent .....                                 | 1676        |
| <b>XIII.</b> | <b>CONCLUSION .....</b>                           | <b>1677</b> |
| <b>XIV.</b>  | <b>RESERVATION OF RIGHTS .....</b>                | <b>1678</b> |



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****TABLES OF EXHIBITS**

| <b>Ex.</b> | <b>'730 Patent Chart</b>   |
|------------|--|
| A-1        | U.S. Patent No. 6,460,070 ("Turek")  |
| A-2        | Distributed Management with Mobile Components ("Kasteleijn")   |
| A-3        | GB 2 363 284 ("Goldman '284")  |
| A-4        | U.S. Patent No. 6,671,724 ("Pandya")   |
| A-5        | U.S. Patent No. 6,012,152 ("Douik ")   |
| A-6        | Proactive Management of Computer Networks using Artificial Intelligence Agents and Techniques ("da Rocha")                 |
| A-7        | U.S. Patent No. US 6,584,502 ("Natarajan")   |
| A-8        | Cisco NetRanger  |
| A-9        | Cisco Catalyst 5000 Switch Family  |
| A-10       | [REDACTED]   |
| A-11       | Cisco Receive ACL ("rACL")   |
| A-12       | Proactive Management of Software Aging ("Castelli")  |
| A-13       | INCA: An Agent-based Network Control Architecture ("INCA")   |
| A-14       | Distributed Network Security Management Using Intelligent Agents ("Boudaoud '98")  |
| A-15       | U.S. Patent No. 5,655,081 ("Bonnell")  |
| A-16       | U.S. Patent No. 6,122,664 ("Boukobza '664")  |
| A-17       | Network Security Management with Intelligent Agents - A First Step with SLD ("Conti")                                      |
| A-18       | U.S. Patent No. 6,665,262 ("Lindskog '262")  |
| A-19       | Network Security Management with Intelligent Agents ("NSMIA")  |
| A-20       | U.S. Patent No. 6,675,128 ("Hellerstein")  |
| A-21       | Distributed Policy Based Management Enabling Policy Adaptation on Monitoring Using Active Network Technology ("Yoshihara") |
| A-22       | U.S. Patent No. 6,826,150 ("Bhattacharya")   |
| A-23       | U.S. Patent No. 6,219,719 ("Graf '719")  |
| A-24       | U.S. Pat. No. 6,243,667 ("Kerr '667")  |
| A-25       | Cisco NetFlow2   |

| <b>Ex.</b> | <b>'659 Patent Chart</b>   |
|------------|--|
| B-1        | U.S. Patent No. 6,460,070 ("Turek")  |
| B-2        | Distributed Management with Mobile Components ("Kasteleijn")   |
| B-3        | GB 2 363 284 ("Goldman '284")  |
| B-4        | U.S. Patent No. 6,671,724 ("Pandya")   |
| B-5        | U.S. Patent No. 6,012,152 ("Douik ")   |
| B-6        | Proactive Management of Computer Networks using Artificial Intelligence Agents and Techniques ("da Rocha") |
| B-7        | U.S. Patent No. US 6,584,502 ("Natarajan")   |
| B-8        | Cisco NetRanger  |
| B-9        | Cisco Catalyst 5000 Switch Family  |
| B-10       | [REDACTED]   |
| B-11       | Cisco Receive ACL ("rACL")   |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

| <b>Ex.</b> | <b>'659 Patent Chart</b>   |
|------------|--|
| B-12       | Proactive Management of Software Aging ("Castelli")  |
| B-13       | INCA: An Agent-based Network Control Architecture ("INCA")   |
| B-14       | Distributed Network Security Management Using Intelligent Agents ("Boudaoud '98")  |
| B-15       | U.S. Patent No. 5,655,081 ("Bonnell")  |
| B-16       | U.S. Patent No. 6,122,664 ("Boukobza '664")  |
| B-17       | Network Security Management with Intelligent Agents - A First Step with SLD ("Conti")                                      |
| B-18       | U.S. Patent No. 6,665,262 ("Lindskog '262")  |
| B-19       | Network Security Management with Intelligent Agents ("NSMIA")  |
| B-20       | U.S. Patent No. 6,675,128 ("Hellerstein")  |
| B-21       | Distributed Policy Based Management Enabling Policy Adaptation on Monitoring Using Active Network Technology ("Yoshihara") |
| B-22       | U.S. Patent No. 6,826,150 ("Bhattacharya")   |
| B-23       | U.S. Patent No. 6,219,719 ("Graf '719")  |
| B-24       | U.S. Pat. No. 6,243,667 ("Kerr '667")  |
| B-25       | Cisco NetFlow2   |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**I. BACKGROUND AND QUALIFICATIONS**

1. I am currently a Professor in the Department of Computer Science at the University of California, Santa Barbara. I also hold an appointment at and am a founding member of the Computer Engineering (CE) Program. I am a founding member of the Media Arts and Technology (MAT) Program, and the Technology Management Program (TMP). I also served as the Associate Director of the Center for Information Technology and Society (CITS) from 1999 to 2012. I have been a faculty member at UCSB since July 1997.

2. I have been asked to submit this report on behalf of Defendant Cisco Systems, Inc. ("Cisco") in the above-captioned litigation. I have been retained as a technical expert by Cisco to study and provide my opinions on the technology that appear in the asserted claims of U.S. Patent Nos. 7,747,730 (the "'730 Patent") and 9,663,659 (the "'659 Patent") (collectively, "the Asserted Patents").

**A. Background**

3. I hold three degrees from the Georgia Institute of Technology: (1) a Bachelor of Science degree in Information and Computer Science (with minors in Economics, Technical Communication, American Literature) earned in June 1992; (2) a Master of Science degree in Computer Science (with specialization in Networking and Systems) earned in June 1994; and (3) a Doctor of Philosophy (Ph.D.) degree in Computer Science (Dissertation Title: Networking and System Support for the Efficient, Scalable Delivery of Services in Interactive Multimedia System, minor in Telecommunications Public Policy) earned in June 1997.

4. One of the major themes of my research has been the delivery of multimedia content and data between computing devices and users. In my research, I have looked at large-scale content delivery systems and the use of servers located in a variety of geographic locations to provide scalable delivery to hundreds, even thousands, of users simultaneously. I have also looked at smaller-scale content delivery systems in which content, including interactive communication like voice and video data, is exchanged between computers and portable computing devices. As a broad theme, my work has examined how to exchange content more

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

efficiently across computer networks, including the devices that switch and route data traffic. More specific topics include the scalable delivery of content to many users, mobile computing, satellite networking, delivering content to mobile devices, and network support for data delivery in wireless network.

5. Beginning in 1992, when I started graduate school, the first focus of my research was on the provision of interactive functions (VCR-style functions like pause, rewind, and fast-forward) for near video-on-demand systems in cable systems, in particular, how to aggregate requests for movies at a cable head-end and then how to satisfy a multitude of requests using one audio/video stream broadcast to multiple receivers simultaneously. Continued evolution of this research has resulted in the development of new techniques to scalably deliver on-demand content, including audio, video, web documents, and other types of data, through the Internet and over other types of networks, including over cable systems, broadband telephone lines, and satellite links.

6. An important component of my research from the very beginning has been investigating the challenges of communicating multimedia content between computers and across networks. Although the early Internet was designed mostly for text-based non-real time applications, the interest in sharing multimedia content quickly developed. Multimedia-based applications ranged from downloading content to a device to streaming multimedia content to be instantly used. One of the challenges was that multimedia content is typically larger than text-only content, but there are also opportunities to use different delivery techniques since multimedia content is more resilient to errors. I have worked on a variety of research problems and used a number of systems that were developed to deliver multimedia content to users.

7. In 1994, I began to research issues associated with the development and deployment of a one-to-many communication facility (called "multicast") in the Internet (first deployed as the Multicast Backbone, a virtual overlay network supporting one-to-many communication). Some of my more recent research endeavors have looked at how to use the scalability offered by multicast to provide streaming media support for complex applications like distance learning, distributed collaboration, distributed games, and large-scale wireless communication. Multicast

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

has also been used as the delivery mechanism in systems that perform local filtering (i.e., sending the same content to a large number of users and allowing them to filter locally content in which they are not interested).

8. Starting in 1997, I worked on a project to integrate the streaming media capabilities of the Internet together with the interactivity of the web. I developed a project called the Interactive Multimedia Jukebox (IMJ). Users would visit a web page and select content to view. The content would then be scheduled on one of a number of channels, including delivery to students in Georgia Tech dorms delivered via the campus cable plant. The content of each channel was delivered using multicast communication.

9. In the IMJ, the number of channels varied depending on the capabilities of the server including the available bandwidth of its connection to the Internet. If one of the channels was idle, the requesting user would be able to watch their selection immediately. If all channels were streaming previously selected content, the user's selection would be queued on the channel with the shortest wait time. In the meantime, the user would *See* what content was currently playing on other channels, and because of the use of multicast, would be able to join one of the existing channels and watch the content at the point it was currently being transmitted.

10. The IMJ service combined the interactivity of the web with the streaming capabilities of the Internet to create a jukebox-like service. It supported true Video-on-Demand when capacity allowed, but scaled to any number of users based on queuing requested programs. As part of the project, we obtained permission from Turner Broadcasting to transmit cartoons and other short-subject content. We also attempted to connect the IMJ into the Georgia Tech campus cable television network so that students in their dorms could use the web to request content and then view that content on one of the campus's public access channels.

11. More recently, I have also studied issues concerning how users choose content, especially when considering the price of that content. My research has examined how dynamic content pricing can be used to control system load. By raising prices when systems start to become overloaded (i.e., when all available resources are fully utilized) and reducing prices when system

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

capacity is readily available, users' capacity to pay as well as their willingness can be used as factors in stabilizing the response time of a system. This capability is particularly useful in systems where content is downloaded or streamed to users on-demand.

12. As a parallel research theme, starting in 1997, I began researching issues related to wireless devices. In particular, I was interested in showing how to provide greater communication capability to "lightweight devices," i.e., small form-factor, resource-constrained (e.g., CPU, memory, networking, and power) devices.

13. Starting in 1998, I published several papers on my work to develop a flexible, lightweight, battery-aware network protocol stack. The lightweight protocols we envisioned were similar in nature to protocols like Universal Plug and Play (UPnP) and Digital Living Network Alliance (DLNA).

14. From this initial work, I have made wireless networking—including ad hoc and mesh networks and wireless devices—another one of the major themes of my research. One topic includes developing applications for mobile devices, for example, virally exchanging and tracking "coupons" through "opportunistic contact" (i.e., communication with other devices coming into communication range with a user). Other topics include building network communication among a set of mobile devices unaided by any other kind of network infrastructure. Yet another theme is monitoring wireless networks, in particular different variants of IEEE 802.11 compliant networks, to (1) understand the operation of the various protocols used in real-world deployments, (2) use these measurements to characterize use of the networks and identify protocol limitations and weaknesses, and (3) propose and evaluate solutions to these problems.

15. As an important component of my research program, I have been involved in the development of academic research into available technology in the market place. One aspect of this work is my involvement in the Internet Engineering Task Force (IETF) including many content delivery-related working groups like the Audio Video Transport (AVT) group, the MBone Deployment (MBONED) group, Source Specific Multicast (SSM) group, the Inter-Domain Multicast Routing (IDMR) group, the Reliable Multicast Transport (RMT) group, the Protocol

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Independent Multicast (PIM) group, etc. I have also served as a member of the Multicast Directorate (MADDOGS), which oversaw the standardization of all things related to multicast in the IETF. Finally, I was the Chair of the Internet2 Multicast Working Group for seven years.

16. I am an author or co-author of nearly 200 technical papers, published software systems, IETF Internet Drafts and IETF Request for Comments (RFCs).

17. My involvement in the research community extends to leadership positions for several journals and conferences. I am the co-chair of the Steering Committee for the ACM Network and System Support for Digital Audio and Video (NOSSDAV) workshop and on the Steering Committees for the International Conference on Network Protocols (ICNP), ACM Sigcomm Workshop on Challenged Networks (CHANTS), and IEEE Global Internet (GI) Symposium. I have served or am serving on the editorial boards of IEEE/ACM Transactions on Networking, IEEE Transactions on Mobile Computing, IEEE Transactions on Networks and System Management, IEEE Network, ACM Computers in Entertainment, AACE Journal of Interactive Learning Research (JILR), and ACM Computer Communications Review.

18. I have co-chaired a number of conferences and workshops including the IEEE International Conference on Network Protocols (ICNP), ACM International Conference on Next Generation Communication (CoNext), IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), International Conference on Communication Systems and Networks (COMSNETS), IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS), the International Workshop On Wireless Network Measurement (WinMee), ACM Sigcomm Workshop on Challenged Networks (CHANTS), the Network Group Communication (NGC) workshop, and the Global Internet Symposium; and I have been on the program committee of numerous conferences.

19. Furthermore, in the courses I teach, the class spends significant time covering all aspects of the Internet including each of the layers of the Open System Interconnect (OSI) protocol stack commonly used in the Internet. These layers include the physical and data link layers and their handling of signal modulation, error control, and data transmission. I also teach DOCSIS,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

DSL, and other standardized protocols for communicating across a variety of physical media including cable systems, telephone lines, wireless, and high-speed Local Area Networks (LANs). I teach the configuration and operation of switches, routers, and gateways including routing and forwarding and the numerous respective protocols as they are standardized and used throughout the Internet. Topics include a wide variety of standardized Internet protocols at the Network Layer (Layer 3), Transport Layer (Layer 4), and above.

20. In addition to having co-founded a technology company myself, I have worked for, consulted with, and collaborated with companies such as IBM, Hitachi Telecom, Digital Fountain, RealNetworks, Intel Research, Cisco Systems, and Lockheed Martin.

21. I am a Member of the Association of Computing Machinery (ACM) and a Fellow of the Institute of Electrical and Electronics Engineers (IEEE).

**B. Previous Expert Witness Experience**

22. My curriculum vitae ("CV") is attached hereto as Appenndix I and provides an accurate identification of my background and experience. It includes a list of all publications I have authored since 1994. It also includes a list of all other cases, since 2014, in which I testified as an expert at trial or by deposition.

**C. Compensation**

23. I am being compensated for services provided in this case at my usual and customary rate of \$650/hour. My compensation is not conditioned on the conclusions I reach as a result of my analysis or on the outcome of this case.

**II. MATERIALS CONSIDERED**

24. In forming the opinions set forth in this report, I have relied on my own personal knowledge and experience, including my experience in the design, development, and operation of relevant systems, as well as my review of the '730 and '659 Patents, their prosecution histories, NetFuel's infringement contentions, NetFuel's amended infringement contentions, and documents describing the Accused Products. A complete listing of all of the documents that I have relied upon in producing this report is provided as Appendix II to this report.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**III. SUMMARY OF OPINIONS**

25. I have been asked to provide opinions regarding whether the claims of the '659 and '730 Patents are invalid as anticipated or would have been obvious to a person having ordinary skill in the art at the time of the alleged invention. For all the reasons detailed below, the claims of the '659 and '730 Patents are anticipated and/or rendered obvious by Turek, Kasteleijn, Goldman, Pandya, Douik, da Rocha, Natarajan, Cisco NetRanger, Cisco Catalyst 5000 Switch Family, [REDACTED], Cisco Receive ACL ("rACL"), Castelli, IBM xSeries Server Software Rejuvenation Agent, INCA, Boudaoud, Bonnell, Boukobza, Conti, Lindskog, NSMIA, Hellerstein, Yoshihara, Bhattacharya, Graf, Kerr, and Cisco NetFlow. It is also my opinion that secondary considerations of non-obviousness do not exist. In addition, the claims of the '659 and '730 Patents lack an adequate written description in the specification, are not enabled, and are indefinite.

**IV. LEGAL STANDARDS**

**A. Anticipation and Obviousness**

26. I understand that a claim is invalid if it is anticipated or obvious. Anticipation of a claim requires that every element of a claim be disclosed expressly or inherently in a single prior-art reference, arranged in the prior-art reference as arranged in the claim. Obviousness of a claim requires that the claim be obvious from the perspective of a person having ordinary skill in the relevant art at the time of the alleged invention. I understand that a claim may be obvious in view of a combination of two or more prior-art references. I also understand that obviousness analysis requires an understanding of the scope and content of the prior art, any differences between the alleged invention and the prior art, and the level of ordinary skill in evaluating the pertinent art.

27. I understand that certain factors—often called “secondary considerations”—may support or rebut the obviousness of a claim. Such secondary considerations include, among other things, commercial success of the alleged invention, skepticism of those having ordinary skill in the art at the time of the alleged invention, unexpected results of the alleged invention, any long-

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

felt but unsolved need in the art that was satisfied by the alleged invention, the failure of others to make the alleged invention, praise of the alleged invention by those having ordinary skill in the art, and copying of the alleged invention by others in the field. I further understand that there must be a nexus—a connection—between any secondary considerations and the alleged invention. I also understand that contemporaneous and independent invention by others is a secondary consideration tending to show obviousness.

28. I further understand that a claim is obvious if it unites old elements with no change to their respective functions, or alters prior art by mere substitution of one element for another known in the field, and that combination yields predictable results. While it may be helpful to identify a reason for this combination, common sense should guide, and there is no rigid requirement for a teaching, suggestion, or motivation to combine. When a product is available, design incentives and other market forces can prompt variations of it, either in the same field or different one. If a person having ordinary skill in the relevant art can implement a predictable variation, obviousness likely bars patentability. Similarly, if a technique has been used to improve one device, and a person having ordinary skill in the art would recognize that the technique would improve similar devices in the same way, use of the technique is obvious. I further understand that a claim may be obvious if common sense directs one to combine multiple prior art references or add missing features to reproduce the alleged invention recited in the claims.

**B. The “Written Description” Requirement**

29. I understand that under 35 U.S.C. § 112(a), claims of a patent are invalid if they fail to provide adequate written description of the claimed invention, including the process and manner of making and using it. This is so that the scope of the right to exclude set forth in the patent claims does not overreach the scope of the inventor's contribution to the field of art as described in the patent specification.

30. In determining whether a patent meets this requirement, I understand one must account for the level of ordinary skill in the art at the time of the priority date of the patent. I further understand that there is an objective standard to determine compliance with the written

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

description requirement. To satisfy the requirement, a patent's specification must convey with reasonable clarity to those of ordinary skill in the art that, as of the asserted priority date, the inventor(s) was/were in possession of the claimed invention.

**C. The "Enablement" Requirement**

31. I understand that under the first paragraph of 35 U.S.C. § 112, a patent's specification must describe the claimed invention in such full, clear, concise, and exact terms as to enable one of ordinary skill in the art to make and use the full scope of the claimed invention. I also understand that in determining whether a patent meets this requirement, one takes into account the level of ordinary skill in the art as of the priority date of the patent. I further understand that a claim is invalid if it is not supported by an enabling disclosure.

32. It is also my understanding that the fact that some experimentation may be required for one of ordinary skill in the art to practice the claimed invention does not necessarily mean the patent fails meet the enablement requirement. However, it is my understanding that factors that may be considered in determining whether undue experimentation would be required to make and use the claimed invention include: (1) the quantity of experimentation necessary; (2) the amount of direction or guidance disclosed in the patent; (3) the presence or absence of working examples in the patent; (4) the nature of the invention; (5) the state of the prior art; (6) the relative skill of those in the art; (7) the predictability of the art; and (8) the breadth of the claims. It is further my understanding that patents are written for persons of skill in the field of the invention. Thus, a patent need not expressly state information that skilled persons would be likely to know or could obtain.

33. I further understand that the full scope of the claims must be enabled, such that the scope of the claimed invention is not greater than the scope of the invention enabled by the specification. I also understand that elements of a claim may not be enabled when the inventor was not able to implement the elements at the time of filing. It is also my understanding that a patent can lack enablement of a given claim element by expressly teaching away from it.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**D. Definiteness**

34. I understand that a patent specification must conclude with one or more claims particularly pointing out and distinctly claiming the subject matter that the applicant regards as his invention. I further understand that claims are indefinite if they do not reasonably apprise those of ordinary skill in the art of the applicant's intended scope of the invention when read in light of the specification. I also understand that in order to show indefiniteness, the party challenging validity must prove so by clear and convincing evidence.

**E. Inventorship**

35. I understand that a correct inventorship is relevant to patent validity and enforceability. I understand that a patent must be filed in the name of all of the inventors of the claimed invention. I understand that nonjoinder is an inventorship error that occurs when the patent erroneously omits a person who is an inventor, and when such an error is not corrected or is uncorrectable, the patent is invalid. I also understand that inventorship errors may also render the patent unenforceable. I further understand the test for inventorship is whether a party had some conceptual role in at least an important or a necessary element, or important and necessary claim.

**F. Subject Matter Eligibility**

36. It is my understanding that a patent claim is invalid if it is directed toward patent-ineligible subject matter. I further understand that determining whether a claim is directed towards a patent-ineligible abstract idea is a two-step process. It must be determined whether the claim is directed towards an abstract idea. If so, it must then be determined whether the claim contains an inventive concept sufficient to transform the claimed abstract idea into a patent-eligible application of that idea. My understanding is that implementing an abstract idea using well-known computer components or functions or taking an abstract idea and adding well-understood, routine, and conventional activities is not sufficient.

**G. Level of Ordinary Skill in the Art**

37. I understand that there are multiple factors relevant to determining the level of ordinary skill in the pertinent art, including the educational level of active workers in the field at

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the time of the alleged invention, the sophistication of the technology, the type of problems encountered in the art, and the prior art solutions to those problems.

38. My conclusion is that a person of ordinary skill in the art (“POSITA”) in the field of the ’659 Patent and ’730 Patent would be a person with the equivalent of a bachelor’s degree in computer science, computer engineering, with relevant experience or exposure to software development for networking applications. In addition, a POSITA would have two to three years of professional experience in software development of networking applications or systems. This educational and professional background would have been necessary to read and understand the ’659 Patent. A person with more graduate education and less experience would also meet the standard, as would a person with significant experience in the field but less education.

**V. BACKGROUND OF THE TECHNOLOGY AND ASSERTED PATENTS****A. The Asserted Patents Concern Automated Management of Large Enterprise Networks**

39. Communication networks allow devices to communicate with one another across a network of interconnected devices. A modern network in a large enterprise, such as a college campus or large corporate network, typically includes multiple different network devices, such as routers and switches to provide network infrastructure (e.g., layer 2 switching or layer 3 routing), or special-purpose network appliances such as firewalls or devices providing network address translation (NAT). Depending on the size of the enterprise network, there may be thousands of network devices in the network, each requiring operations, administration, and management (OAM). In the mid-to-late 1990s, network devices were frequently manually maintained by a network administrator or IT professional that would perform OAM, such as updating the device policies or troubleshooting, via a network device’s command line interface (CLI), either by plugging into a management port, or through the use of a remote telnet login. Additionally, because network equipment manufacturers had yet to standardize their individual CLIs, OAM was a significant cost center for network operators.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****B. The Asserted Patents Describe a Hierarchical Network Management System**

40. The Asserted Patents share a common specification, and are both titled “Managing Computer Network Resources.” The background of the Asserted Patents states “[c]omputer networks need to be constantly managed in order to ensure smooth and efficient operation” and that “[t]ypically, network management is performed by humans or is, to a large extent, dependent on human input.” ’730 Patent at 1:10–23; ’659 Patent at 1:14–26. The Asserted Patents describe this human involvement as “undesirable, particularly, for example, in the case of a network having a large number of nodes because of the time it would take a human to identify and fix a failed node” and state that it is “desirable that networks run themselves as much as possible.” ’730 Patent at 1:10–23; ’659 Patent at 1:14–26.

41. The Asserted Patents are organized much like a high-level research whitepaper. Figure 1 and the text under the heading “System Overview” (’730 Patent at 2:21-4:2) lays out the overall architecture of the purported invention. The required characteristics of each component of the system are then described under a heading for each component, with references to the figures. For example, under the heading “The ARE”, the patent discusses the components of the agent runtime environment (’730 Patent at 4:3–5:31, Figs. 2 and 3), before discussing particular operations such as ARE startup (’730 Patent at 5:32–6:51, Fig. 4), and agent startup (’730 Patent at 6:52–7:19, Fig. 5). Similarly, under the heading “Agents,” the patent defines agents of the purported invention, describing their proposed structure and characteristics under the subheadings “Agent Structure” and “General Characteristics of Agents,” respectively. While several of these components contain text describing particular embodiments, due to the structure of the Asserted Patents, they are clearly delineated with leading text. Two CORBA-based implementation examples are described near the end of the patent under the heading “Specific Implementation” (’730 Patent at 28:7–29:5), and the end of the patent a section heading “EXAMPLES” provides “[s]pecific examples of system 10 components” such as high-level class definitions of the messaging adapter 112. ’730 Patent at 30:16–34:54.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

42. The purported inventive architecture of the Asserted Patents is a “vertically layered” hierarchy. *See* ’730 Patent at 2:21–46, Fig. 1; *see also* ’730 Patent at Fig. 1, 1:58–59 (“FIG. 15 shows the hierarchy of the various modelers within the system.”). Each layer of the system 10 masks the complexity of the layers below it and may be configured to provide guaranteed services to the layers above it. *See* ’730 Patent at 2:21–46, Fig. 1. The Asserted Patents state “[t]he upper most layer of the system 10 is occupied by an agent control mechanism 16 which comprises a CCE<sup>+</sup> layer 18 sitting on top of a CCE<sup>-</sup> layer 20.” ’730 Patent at 2:29–31. The Asserted Patents disclose that for “policy handling and distribution within the system 10, the CCE<sup>+</sup> 18 may be regarded as a global modeler [and] the CCE<sup>-</sup> 20 as a regional modeler.” ’730 Patent at 3:30–34. With reference to Figure 1, the upper most layer of the system 10 is occupied by an agent control mechanism 16, which itself has two layers, a global modeler hierarchically above a regional modeler. *See* ’730 Patent at 2:29–40.

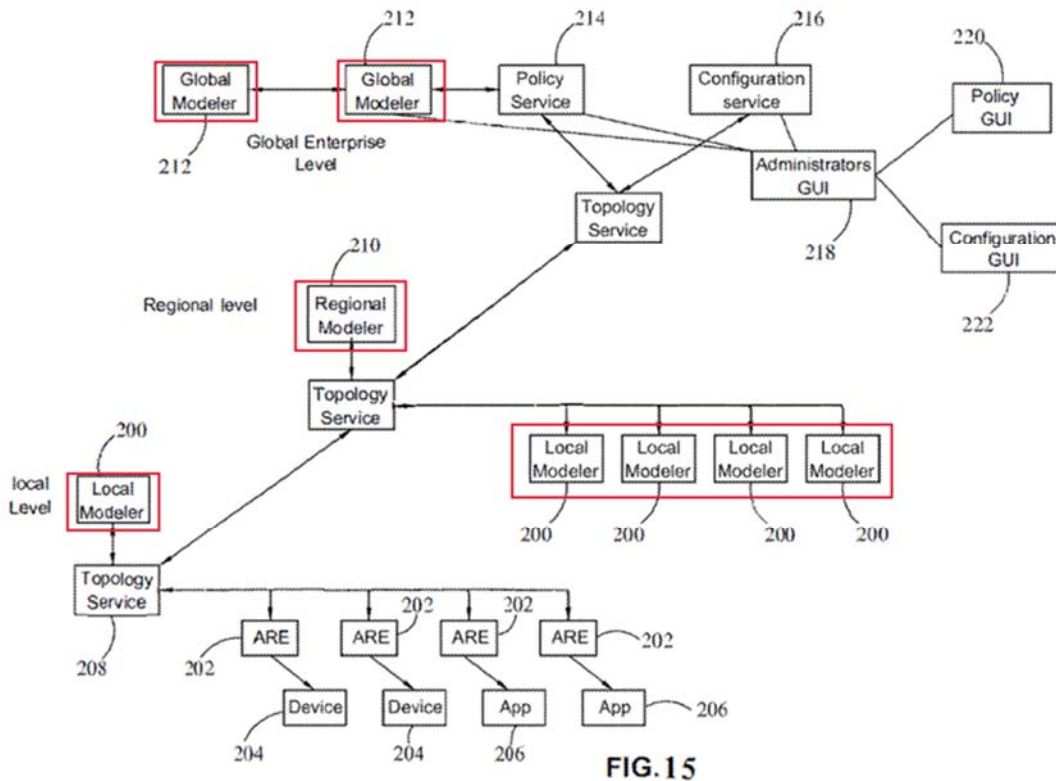
43. The Asserted Patents state that “[b]ecause the network on which system 10 is installed is large, it is preferable to divide the network into a number of functional/management domains.” *See* ’730 Patent at 3:10–16. The CCE<sup>-</sup> 20 (i.e. the regional modeler) “may be viewed as a control mechanism for a particular functional domain.” *See* ’730 Patent at 3:16–20.

44. Each functional domain is comprised of a number of network devices (called “host platform 14” in the Asserted Patent), each loading one or more “agent runtime environments,” or ARE 12. *See* ’730 Patent at 2:20–25. The Asserted Patents explain that “[t]ypically, the system 10 is implemented on a large network comprising a number of network devices, each hosting an ARE. The AREs are able to create/instantiate, suspend operation of the agents, and terminate the agents.” *See* ’730 Patent at 2:47–50. Each device also includes a “domain adapter 15 which facilitates communication between the CCE<sup>-</sup> 20 [regional modeler] and the AREs in a particular functional domain.” *See* ’730 Patent at 3:24–27.

45. Hierarchically below the “regional modelers” are “local modelers.” *See, e.g.,* ’730 Patent at Fig. 15. The Asserted Patents disclose that for “the combination of each domain adapter 15 and its associated ARE 12 as a local modeler.” ’730 Patent at 3:30–34. The system as described

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

is shown below in Figure 15, which “shows the hierarchy of the various modelers within the system.” ’730 Patent at 1:58–59.



46. At the lowest level of the hierarchy are the individual software agents that execute policies. See ’730 Patent at Fig. 15, 6:33–35. By utilizing this hierarchical architecture, policy updates may be pushed down from a global modeler, regional modeler, to individual local modelers, and then updated on a set of agents, thereby reducing the amount of human interaction required to administer the network. See ’730 Patent at 20:7–22. The purported invention also claims the ability for events detected by agents to be “passed up” the hierarchy—that is, an agent may encounter a problem it cannot solve, pass it up to the local modeler, which determines how to solve the problem and provides a policy update to the agent, or, alternatively, cannot solve the problem and passes it up to the regional modeler, and so on. See ’730 Patent at 3:55–62. This hierarchical structure, where the “[p]olicy is distributed in the system 10 along a chain defined by the global, regional and the local modelers” and the “global modeler breaks up and enforces policies upon subordinate modelers” is specifically intended to “add scalability and robustness and



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to reduce decision-making time.” *See* ’659 Patent at 20:20–24. Moreover, the hierarchical system “implements a competence based control flow mechanism wherein, in responding to a situation which requires corrective action, an agent will first attempt to take the appropriate corrective action, should it have the appropriate corrective policy. If the agent lacks the corrective policy it will request it from its host ARE, or other agents within its functional domain, or the CCE- 20, or the CCE- 18, as the case may be, and in the particular order as recited above.” ’659 Patent at 3:61–4:3. The defined structure and capabilities of the various components of the invention are necessary to carry out these operations, which are claimed in the independent claims of the Asserted Patents.

47. For example, the ’730 Patent claims are largely directed to determining an “optimal policy” and dynamically updating the agents with this policy. *See* ’730 Patent at claim 1. The specification states that various information is collected by the agents and delivered to the global modeler to model network behavior. *See* ’730 Patent at 3:1–10. The global modeler includes a module 18.3 that “allows test policy to be written and the effects of such test policy on the system 10 may be determined by using numerical modeling techniques in the module 18.4. In this way, policy may be tested before implementation in order to determine optimal policy.” ’730 Patent at 3:1–10. After determining this optimal policy, it is dynamically updated on the agents, and the agents begin implementing the “optimal policy.” ’730 Patent at 3:1–10.

48. The structure and capabilities of the various components in the hierarchy of the purported invention are also central to the implementation of the operations claimed in the ’659 Patent. For example, the ’659 Patent claims are directed, in part, to determining whether an agent has a “corrective policy” and, if the agent does not have the “corrective policy,” requesting it from a global modeler so that the received “corrective policy” can be implemented by the agent. *See* ’659 Patent at claim 1, 3:40–4:3. According to the ’659 Patent specification, “agents are used to monitor various operational parameters,” and “each ARE has the ability to provide feedback on the per-thread utilization of each thread running within it.” *See* ’659 Patent at 3:40–42, 11:27–30. An agent “will first determine whether it has the necessary corrective policy to change the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

operational parameters,” and “if the agent lacks the corrective policy then it will request the corrective policy from the particular ARE within which it is operating.” ’659 Patent at 3:46–53. The ARE will attempt to obtain corrective policy from other agents and AREs within the same functional domain and, if unsuccessful, will then make “a request to the CCE<sup>-</sup> 20 for the corrective policy.” ’659 Patent at 3:53–58. The regional modeler “CCE<sup>-</sup> 20 will respond by either supplying the corrective policy to the agent or it would obtain the corrective policy from the CCE<sup>+</sup> 18 [global modeler], if it does not have the corrective policy. ’659 Patent at 3:58–61. The Regional Modeler (CCE<sup>-</sup> 20) includes a “Local Domain Policy Request Module 20.3,” and this “Module 20.3 allows the CCE<sup>-</sup> 20 to request domain specific [sic] policy from the CCE<sup>+</sup> 18 [global modeler] where appropriate.” ’659 Patent at 3:25–27. The requested policy may be pushed down to and enforced by the agent, which performs a corrective action based on the received corrective policy from the global modeler. *See* ’659 Patent at 3:40–4:3.

**C. Priority Dates**

49. The ’730 Patent issued from U.S. Pat. Appl. No. 10/186,526, filed on June 28, 2002.

50. The ’659 Patent issued from U.S. Pat. Appl. No. 13/648,623, filed on October 12, 2012, is a continuation of application number 13/363,203, filed on January 31, 2012. Accordingly, it is my view that the asserted claims of the ’659 Patent are entitled to a priority date no earlier than January 31, 2012.

51. I understand that NetFuel contends that the priority date for the asserted claims of the ’659 Patent is June 28, 2002. It is my understanding, however, that a priority claim to a parent application must be made during the pendency of the parent application. NetFuel did not do that and did not revive its delayed priority claim. I understand that NetFuel contends that combinations of Control Plane Policing (CoPP), Embedded Event Manager (EEM), and Local Packet Transport Services (LPTS), as well Excessive Punt Flow Trap (EPFT) (collectively, the “Accused Features”) infringe claims of the ’659 Patent. But these Accused Features were in existence and publicly disclosed prior to the priority date of the ’659 Patent. Accordingly, under NetFuel’s interpretation and application of the claims in its infringement contentions, and if those interpretations and

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

applications are correct, the Accused Features are prior art to and anticipate and/or render obvious the claims of the '659 Patent.

**VI. THE ASSERTED CLAIMS**

52. I understand that NetFuel has asserted Claims **1**, 2, 3, 4, 6, **7**, 10, 11, 12, 16, 17, 18, 19, 21, 22, 24, 26, 29, **30**, 31, 32, 33, 34, and 36 of the '730 Patent and claims **1**, 2, 3, 6, **7**, 8, 9, **13**, 14, 15 and 18 of the '659 Patent. The numbers in bold font correspond to independent claims.

**VII. CLAIM CONSTRUCTION**

53. My opinions regarding invalidity of the Asserted Claims are based on these constructions. With regard to claim language not specifically construed, I viewed that language from the perspective of one of ordinary skill in the art at the effective filing date of the patent, as informed by the Asserted Patents, the intrinsic evidence, and extrinsic evidence where appropriate, consistent with the legal standards noted above.

54. The following constructions have been adopted with regard to the '730 and '659 Patents. *See* Dkt. 71 (Claim Construction Order):

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

| <b>Term</b>                    | <b>Patent and Claims</b>                                      | <b>Construction</b>  |
|--------------------------------|---|--|
| clone itself                   | '730 Patent, claims 1, 7, 30                                  | Plain and ordinary meaning.<br>No construction necessary.  |
| computer network               | '730 Patent, claims 1, 7, 30                                  | Plain and ordinary meaning.<br>No construction necessary.  |
| load balancing operation       | '659 Patent, claims 2, 8, 14                                  | Plain and ordinary meaning.<br>No construction necessary.  |
| agent                          | '659 Patent, claims 1, 7, 13                                  | A program that performs some type of operation, which may be information gathering or some processing task, in the background.           |
| software agent                 | '730 Patent, claims 1, 7, 30                                  | Plain and ordinary meaning.<br>No construction necessary.  |
| runtime environment            | '659 Patent, claims 1, 7, 13;<br>'730 Patent, claims 1, 7, 30 | Plain and ordinary meaning.<br>No construction necessary.  |
| topological representation     | '730 Patent, claims 3, 9, 32                                  | A representation of the logical and/or physical arrangement of devices in a network.   |
| secure communications protocol | '730 Patent, claims 12, 13, 14                                | A set of rules or standards designed to protect data from accidental or malicious access, use, modification, destruction, or disclosure. |
| autonomous agent               | '730 Patent, claims 1, 7, 30                                  | Plain and ordinary meaning.<br>No construction necessary.  |
| thread                         | '659 Patent, claims 1, 7, 13                                  | the execution of a section of code independently within a software program   |
| optimal policy                 | '730 Patent, claims 1, 7, 30                                  | a policy that is the most effective for a particular characteristic or set of characteristics  |
| global modeler                 | '659 Patent, claims 1, 7, 13                                  | a module that enforces policies upon subordinate modelers  |

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

| <b>Term</b>  | <b>Patent and Claims</b>     | <b>Construction</b>   |
|--|------------------------------|---|
| first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment | '659 Patent, claims 1, 7, 13 | if the corrective policy is not available to an agent operating within the first runtime environment, first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment |

**VIII. ANTICIPATORY PRIOR ART FOR THE '659 PATENT****A. Anticipation by and/or Obviousness in View of United States Patent No. 6,460,070, filed 6/3/1998 and issued to Turek, et al. on 10/2/2002.**

55. Turek is titled "Mobile Agents for Fault Diagnosis and Correction in a Distributed Computer Environment," has a filing date of June 3, 1998, and issued on October. 1, 2002. As explained in detail below and in the chart attached as Ex. B-1, Turek anticipates and renders obvious the claims of the '659 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***a. *A computer-implemented method, comprising:*

56. Turek discloses the preamble of claim 1, "[a] computer-implemented method, comprising . . . ." '659 Pat. at Cl. 1. For example, Turek discloses a method and apparatus for "managing a large distributed computer enterprise environment and, more particularly, to diagnosing and correcting network faults in such an environment using mobile software agents."

Turek at 1:7–10, 2:26-33, 5:3-19, 9:30-43, 10:28-42, Figs. 1, 3, 5. Further, Turek discloses that:

Referring now to FIG. 1, *the invention is preferably implemented in a large distributed computer environment* 10 comprising up to thousands of "nodes." The nodes will typically be geographically dispersed and the overall environment is "managed" in a distributed manner. Preferably, the managed environment (ME) is logically broken down into a series of loosely-connected managed regions (MR) 12, each with its own management server 14 for managing local resources with the MR. The network typically will include other servers (not shown) for carrying out other distributed network functions.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Turek at 3:47-57.<sup>1</sup> Turek discloses that “[a] method of diagnosing a fault in such an environment begins by deploying a management infrastructure throughout the computer network, the management infrastructure including a runtime environment at each of the endpoint machines.” Turek at Abstract. In particular, “[i]n response to occurrence of the fault, a software agent is selected, the software agent being executable by the runtime environment at an endpoint machine. The selected software agent is then deployed into the computer network to diagnosis the fault.” Turek at Abstract. Turek’s software agents “are available at a central location (e.g., manager 14) or at a plurality of locations,” and “traverse the network to diagnose and, if possible, to correct a network fault.” Turek at 5:33-42.

b. *running at least one thread in a first runtime environment;*

57. Turek discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example, “[t]he management infrastructure includes a dispatch mechanism preferably located at a central location, and a runtime environment supported on given nodes of the network. In particular, the runtime environment (e.g., an engine) is preferably part of a distributed framework supported on each managed node of the distributed enterprise environment.” Turek at 2:30-36. Turek also discloses that:

At each node, the software agent is preferably run by the runtime engine previously deployed there. Alternatively, the software agent runs as a standalone process using existing local resources. As noted above, when the event is a fault, the software agent locates the fault and attempts to rectify it. If necessary, the software agent may obtain additional code from the dispatch mechanism or some other network source. Such additional code may be another software agent.

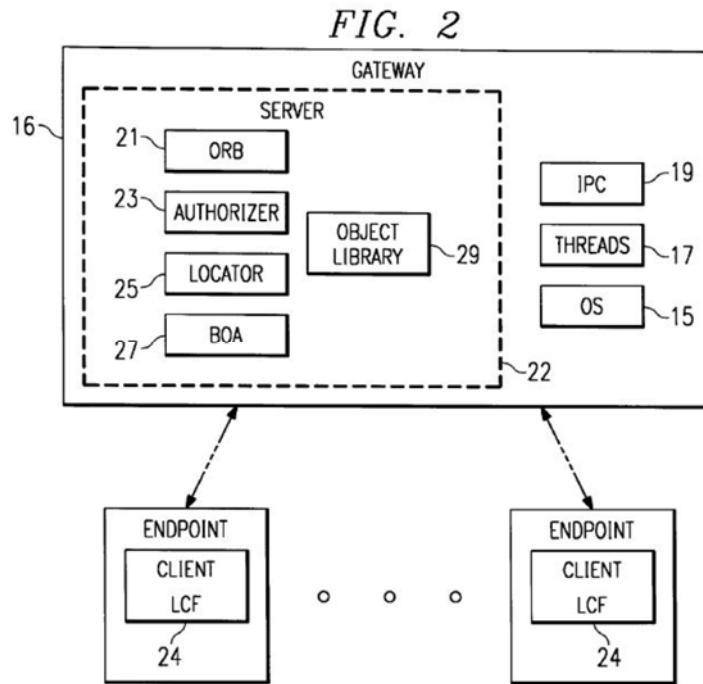
Turek at 2:63-3:3; *see also* 3:67-4:4 (“The server component 22 is a multi-threaded runtime process. . . .”), 5:3-19, 6:49-59, 6:60-67, 9:11-43, 9:65-10:7. Turek also discloses that, in a representative embodiment, “both the runtime engine and the software agent(s) are conveniently written in Java” (6:38-39), and “[a]s is known in the art, Java is an object-oriented, multi-threaded, portable, platform-independent, secure programming environment used to develop, test and

---

<sup>1</sup> All emphases added unless otherwise noted.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

maintain software programs.” Turek at 6:40-44. Figure 2 additionally depicts threads 17 mechanism 17 within the client component of Turek’s management system (3:24-30, 4:10-12).



Turek at Fig. 2.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

58. Turek discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. As described *supra* §VIII.A.1.b, Turek discloses management of a multi-threaded network.

Turek also discloses that:

The runtime engine 41 may also comprise generic diagnostic functionality that may be useful in diagnosing and rectifying errors that arise in the computer network. Preferably, it is expected that the runtime engine interacts with a software agent for this purpose. In a preferred embodiment, the present invention envisions automatic deployment of one or more software agents to locate a particular network fault, and the use of such agent (once the fault is located) to rectify the problem.

Turek at 6:22-30. Further, “the system management framework facilitates execution of system *management tasks required to manage the resources* in the MR [managed region]. Such tasks

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

are quite varied and include, without limitation, file and data distribution, *network usage monitoring*, user management, printer or other resource configuration management.” Turek at 4:22–29; *see also* 4:29–49, 5:3–19, 7:32–47, 8:18–32, 9:21–29, Figs. 4, 5, 6.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

59. Turek discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example, Fig. 4 shows Turek’s fault detection method:

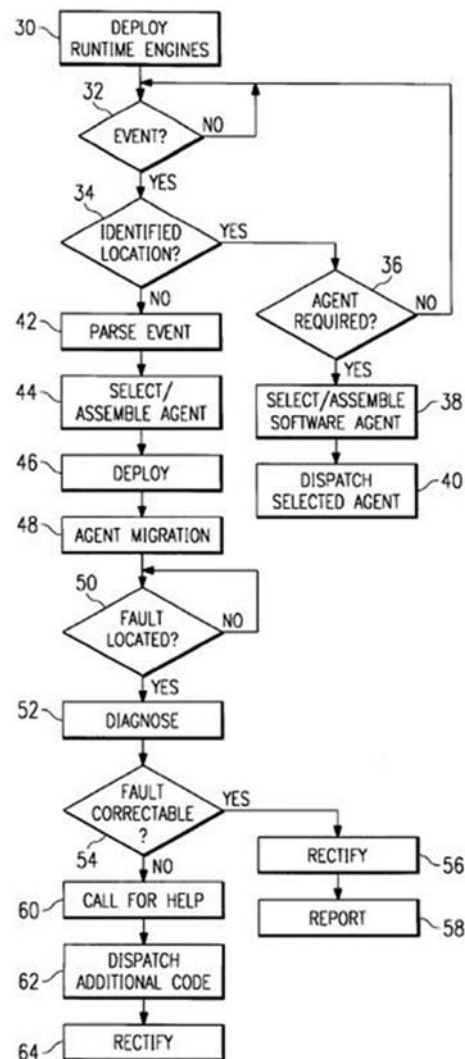


FIG. 4

Turek at Fig. 4.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

60. Turek further explains that:

At step 32, a test is performed at the dispatch mechanism 15 to determine whether a given event has occurred. As noted above, for illustrative purposes, a given event is a network "fault", alarm or other such trigger. One of ordinary skill will appreciate that the particular event need not be a fault or alarm type of condition, however. An alternative event might be a request for maintenance in some non-specific area of the network. The software agent might then be deployed to the general area from which the request originated but then used to identify a specific target location at which the request will be serviced. The dispatch mechanism preferably includes an appropriate user interface to enable an administrator to identify event(s). Alternatively, event consumers may subscribe to the dispatch mechanism by identifying particular events of which they are interested in receiving notice.

Turek at 6:67-7:14.

In particular, at step 42, the event is parsed to determine whether it provides any clue as to where the fault originates. The dispatch mechanism preferably maintains a database (reference numeral 43 in FIG. 5) of information derived from prior events that is useful in determining how a particular event may be diagnosed and corrected. Dispatch mechanism also preferably includes the event "correlator" (or rule base) (reference numeral 45 in FIG. 5) that uses the information in the database to help determine the nature and potential location of the event. As an example, a particular "event" may comprise a plurality of alarms generated by a number of resources in a particular area of the network. By parsing such information through the event correlator/database, the dispatch mechanism may make a decision about the cause of the event. In this sense, the information received by the dispatch mechanism provides a clue regarding how the new event should be addressed.

Turek at 7:32-47; *see also* 2:8-10, 2:33-46, 4:22-28, 5:3-6:4, Figs. 4, 5, 6.

e. *and performing a corrective action to fix any detected abnormalities,*

61. Turek discloses this element of claim 1, "and performing a corrective action to fix any detected abnormalities." '659 Pat. at Cl. 1. For example, Turek discloses that "[o]nce the fault or other problem has been diagnosed, the agent attempts to fix the problem[.] The agent may have the necessary code or it may send requests to the dispatch mechanism for additional code to effect the repair." Turek at 9:21-29. Further, "the present invention envisions automatic deployment of one or more software agents to locate a particular network fault, and the use of such agent (once the fault is located) to rectify the problem." Turek at 6:27-31; *see also* 1:65-67, 2:33-3:3, 5:65-6:4, 6:23-27, 6:31-37, 6:67-7:29, 7:32-48, 7:49-57, Figs. 4, 5, 6, Abstract.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

62. Turek discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example, Turek discloses that “[o]nce the fault or other problem has been diagnosed, ***the agent attempts to fix the problem[.]*** ***The agent may have the necessary code or it may send requests to the dispatch mechanism for additional code to effect the repair,***” and “[t]he additional code may be other software agent(s).” Turek at 9:21-29. Further, Turek explains that “[i]f unable to effect repairs, the agent will, at a minimum, report back with the diagnosis to a user interface of the dispatch mechanism.” Turek at 9:21-29; *see also* 2:66-3:3 (“As noted above, when the event is a fault, the software agent locates the fault and attempts to rectify it. If necessary, the software agent may obtain additional code from the dispatch mechanism or some other network source. Such additional code may be another software agent”), 2:4-7 (“It is yet another object of this invention to select a given software agent from a set of such agents based on a particular fault and to dispatch the selected agent into the network to locate and correct the fault”), 2:29-46, 2:63-3:3, 5:31-42, 6:67-7:14, 7:15-28, 7:33-47, 7:48-57, 7:58-8:9, 8:18-65, 9:10-20, 9:21-29, Figs. 4, 5, 6. Turek discloses that, as shown in Fig. 4, a request is made for corrective policy if the software agent is unable to correct the problem:

If, however, the outcome of the test at step 54 indicates that ***the software engine and/or the runtime engine cannot rectify the problem, a call is made at step 60 to obtain additional help.*** In a preferred embodiment, dispatch mechanism 62 responds to the call and dispatches additional code (e.g., another software engine) to the node to assist in the problem diagnosis and/or correction, as the case may be. ***At step 64, the fault is rectified.*** This completes the processing.

Turek at 8:10-17.

63. As explained above, Turek discloses that the dispatch mechanism is external to the runtime environments at each node of the managed system. *See, e.g.,* Turek at 5:37-40 (“The

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

software agents are 'mobile' in the sense that the agents are dispatched (as will be described below) from a dispatch mechanism and then migrate throughout the network environment.”), 9:10-20, Figs. 4, 5, 6.

In one particular embodiment of the present invention, a software agent is a Java applet (e.g., comprised of a set of Java “class” files) and the runtime environment includes a Java Virtual Machine (JVM) associated with a Web browser. In this illustrative example, various nodes of the network are part of the Internet, an intranet, or some other computer network or portion thereof. When the given fault occurs, *the dispatch mechanism compiles the appropriate Java class files (based on the event) and dispatches the applet (as the software agent) across the network to the target node.* The applet is executed on the JVM located at the target node.

Turek at 6:49-59.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

64. Turek discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. As described above in connection with claim elements 1.e-1.f, Turek discloses that upon detection of an event, “the dispatch mechanism selects a software ‘agent’, preferably from a set of software agents useful in diagnosing network events.” Then, “[t]he software agent selected or created is preferably a set of tasks that are selected or assembled based on the nature of the given event. Thus, the particular triggering ‘event’ is used to provide clues as to the network location to which the agent should be sent, as well as the type of agent to send.” Turek at 2:35-46; *see also* 2:29-46, 2:63-3:3, 5:31-42, 6:67-7:14, 7:15-28, 7:33-47, 7:48-57, 7:58-8:9, 8:18-65, 9:10-20, 9:21-29, Figs. 4, 5, 6. Turek additionally discloses that:

The dispatch mechanism preferably maintains a database (reference numeral 43 in FIG. 5) of information derived from prior events that is useful in determining how a particular event may be diagnosed and corrected. Dispatch mechanism also preferably includes the event "correlator" (or rule base) (reference numeral 45 in FIG. 5) that uses the information in the database to help determine the nature and potential location of the event. As an example, a particular “event” may comprise a plurality of alarms generated by a number of resources in a particular area of the network. By parsing such information through the event correlator/database, the dispatch mechanism may make a decision about the cause of the event. In this sense,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the information received by the dispatch mechanism provides a clue regarding how the new event should be addressed.

Turek at 7:33-47.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

65. Turek discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. As explained above, Turek discloses management of a multi-threaded distributed network using a manager with a dispatch mechanism and mobile software agents. *See supra*, Claim 1, elements b-g. In particular, Turek discloses that:

In this example, the manager 14 includes the dispatch mechanism 35 having a set of software agents 37 associated therewith. Alternatively, dispatch mechanism 35 may include a set of configurable software tasks 39 from which one or more agents are constructed. Manager 14 preferably also includes a database 43 and an event correlator 45, for the purposes described below. The dispatch mechanism itself may be distributed across multiple nodes. Each of the gateway nodes 16 and each of the terminal nodes 18 (or some defined subset thereof) include a runtime engine 41 that has been downloaded to the particular node via a distribution service. The engine 41 provides a runtime environment for the software agent.

Turek at 5:65-6:10.

66. Turek discloses that each runtime environment runs multiple threads. *See* Turek at 3:67-4:4 (“The server component 22 is a multi-threaded runtime process that comprises several components: an object request broker or ‘ORB’ 21, an authorization service 23, object location service 25 and basic object adaptor or ‘BOA’ 27.”); *see also* 6:38-44 (“both the runtime engine and the software agent(s) are conveniently written in Java. As is known in the art, Java is an object-oriented, multi-threaded, portable, platform-independent, secure programming environment used to develop, test and maintain software programs.”). Fig. 5 shows Turek’s management infrastructure implemented in a distributed network with management of multiple nodes, within which terminal nodes are running:

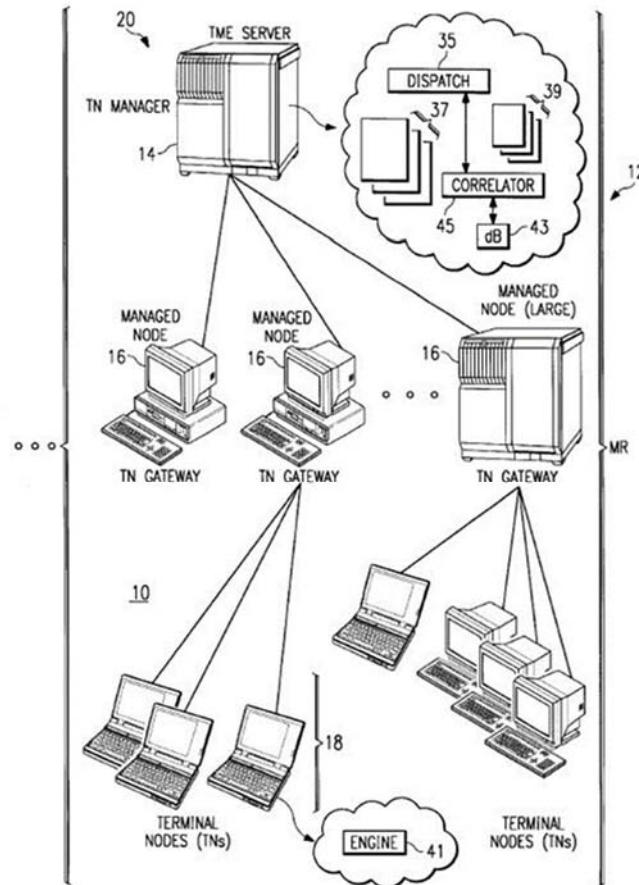
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 5

Turek at Fig. 5.

67. Turek further discloses that the dispatch mechanism maintains global information about the managed nodes, as “[a]t step 58, information about the problem and the corrective action that as undertaken are reported back to the dispatch mechanism and stored in the database for future use.” Turek at 8:1-9; *see also* 2:21-25, 2:63-3:3, 3:65-4:3, 6:49-59, 6:67-7:7, 8:10-17, 8:28-37, 8:43-51, 8:53-58, 9:10-29, Figs. 1, 2, 4, 6.

2. **Claim 2**

a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

68. Turek discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example, Turek discloses that:

Preferably, the managed environment (ME) is logically broken down into a series of loosely-connected managed regions (MR) 12, each with its own management server 14 for managing local resources with the MR. The network typically will include other servers (not shown) for carrying out other distributed network functions. These include name servers, security servers, file servers, threads servers, time servers and the like. ***Multiple servers 14 coordinate activities across the enterprise and permit remote site management and operation.*** Each server 14 serves a number of gateway machines 16, each of which in turn support a plurality of endpoints 18. The server 14 coordinates all activity within the MR using a terminal node manager 20.

Turek at 3:51-64. Moreover, “the system management framework facilitates execution of system management tasks required to manage the resources in the [managed region]. Such tasks are quite varied and include, without limitation, ***file and data distribution, network usage monitoring***, user management, printer or other resource configuration management, and the like.” Turek at 4:13-28.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

69. Turek discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example, Turek discloses comparing event data against a database of event thresholds maintained by the dispatch mechanism:

In particular, at step 42, the event is parsed to determine whether it provides any clue as to where the fault originates. The dispatch mechanism preferably maintains a database (reference numeral 43 in FIG. 5) of information derived from prior events that is useful in determining how a particular event may be diagnosed and corrected. Dispatch mechanism also preferably includes the event “correlator” (or rule base) (reference numeral 45 in FIG. 5) that uses the information in the database to help determine the nature and potential location of the event. As an example, a particular “event” may comprise a plurality of alarms generated by a number of resources in a particular area of the network. By parsing such information

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

through the event correlator/database, the dispatch mechanism may make a decision about the cause of the event. In this sense, the information received by the dispatch mechanism provides a clue regarding how the new event should be addressed.

Turek at 7:32-47.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

70. Turek discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm,” at least for the reasons discussed below and based on NetFuel's interpretation of the claims. '659 Pat. at Cl. 6. For example, as described above (*see* claim 1), Turek discloses a self-healing system with its “primary objective” being “to automatically diagnose faults or other events that occur in a large, distributed computer network.” Turek at 1:65-67; *see also* 3:47-51, 5:48-60, 8:20-33, 3:51-64, 4:41-49, 5:51-60, Fig. 1.

71. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

72. Turek discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” '659 Pat. at Cl. 7. For example,

- b. *running at least one thread in a first runtime environment;*

73. Turek discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

74. Turek discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;and*

75. Turek discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *performing a corrective action to fix any detected abnormalities;*

76. Turek discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

77. Turek discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

78. Turek discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

79. Turek discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

80. Turek discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

81. Turek discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example, “Preferably, the server and each of the gateways is a computer or ‘machine.’ For example, each computer may be a RISC System/6000(a) (a reduced instruction set or so-called RISC-based workstation) running the AIX (Advanced Interactive Executive) operating system, preferably Version 3.2.5 or greater. Suitable alternative machines include: an IBM-compatible PC x86 or higher running Novell UnixWare 2.0, an AT&T 3000 series running AT&T UNIX SVR4MP-RAS Release 2.02 or greater, Data General AViiON series running DG/UX version 5.4R3.00 or greater, an HP9000/700 and 800 series running HP/UX 9.00 through HP/UX 9.05. Motorola 88K series running SVR4 version R40V4.2, a Sun SPARC series running Solaris 2.3 or 2.4, or a Sun SPARC series running SunOS 4.1.2 or 4.1.3. Of course, other machines and/or operating systems may be used as well for the gateway and server machines.” (Turek ’070 at Col. 5:3-17).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

82. Turek discloses this element of claim 13, “a processor,” at least for all the reasons explained above regarding claim 1, element c. ’659 Pat. at Cl. 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

83. Turek discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising,” at least for all the reasons explained above regarding claim 1, element c. ’659 Pat. at Cl. 13.

d. *running at least one thread in a first runtime environment;*

84. Turek discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

85. Turek discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

86. Turek discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

87. Turek discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective policy is not available to an agent operating within the first runtime environment,*

88. Turek discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

89. Turek discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

90. Turek discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

91. Turek discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*known thresholds.*

92. Turek discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

93. Turek discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

94. I expect to testify that Turek anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-2.

**B. Anticipation by and/or Obviousness in View of Great Britain Patent No. 2 363 284, issued to Goldman, et al. on 12/12/2001.**

95. As explained in detail below and in the chart attached as Ex. B-3, Goldman anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

a. *A computer-implemented method, comprising:*

96. Goldman discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

1. ***A computer implemented method*** for deploying a policy [120] to a target [110], comprising the steps of: assigning the policy [120] to the target [110], providing the policy [120] specifies conditional action implementable on the target [110], providing ***the target is a resource on a network [220]***, and providing policy [120] assignment comprises association of the policy [120] with the target [110] prior to policy [120] reconfiguration of the target [110]; and activating the policy [120] on the target [110], providing the policy [120] has been activated when target [110] actions comply with the policy [120].

Goldman at Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

97. Further examples include Goldman at p. 1, lns. 7-8; p.6, lns. 4-9; p. 13, lns. 14-20; p. 13, lns. 21-28; Figs. 4A-4B.

b. *running at least one thread in a first runtime environment;*

98. Goldman discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

In two stage commitment, a first stage comprises *the programming of the policy into the target or onto a policy configuration agent*, while a second stage comprises the activation of the policy on the target. Prior to activation the policy resides on the target or on the policy configuration agent but is not active in the operation of the target.

Goldman at p. 5, lns. 2-6.

99. Further examples include Goldman at p. 6, lns. 4-9.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

100. Goldman discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

In two stage commitment, a first stage comprises the programming of the policy into the target or onto a policy configuration agent, while a second stage comprises the activation of the policy on the target. Prior to activation the policy resides on the target or on the policy configuration agent but is not active in the operation of the target.

Goldman at p. 5, lns. 2-6.

101. To the extent this limitation is not expressly disclosed by Goldman, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

d. *detecting if there is an abnormality in the monitored operational parameters;*

102. Goldman discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The present patent document relates to a novel method for deployment of policy to a target connected to a network for the purpose of controlling the actions of that target based upon certain predefined conditions. In representative embodiments, methods are disclosed for creating another step in the policy deployment process within which ***policies can be created, tested, changed, and deleted*** prior to their transfer to the policy configuration agents of the targets to which it is intended that they will eventually be deployed.

Goldman at p. 4, lns. 3-9.

103. Further examples include Goldman at p. 2, ln 29 - p.3, ln. 3, p. 16, lns. 17-25; Goldman at fig. 7.

- e. *and performing a corrective action to fix any detected abnormalities,*

104. Goldman discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example,

The present patent document relates to a novel method for deployment of policy to a target connected to a network for the purpose of controlling the actions of that target based upon certain predefined conditions. In representative embodiments, methods are disclosed for creating another step in the policy deployment process within which ***policies can be created, tested, changed, and deleted*** prior to their transfer to the policy configuration agents of the targets to which it is intended that they will eventually be deployed.

Goldman at p. 4, lns. 3-9.

105. Further examples include Goldman at p. 2, ln 29 - p.3, ln. 3, p. 16, lns. 17-25; Goldman at Fig. 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

106. Goldman discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Providing an assignment stage ***allows users to make an association between a policy and the policy enforcement point, or target***, without affecting or committing to changing the active policy on that target. Note

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

that two stage deployment is independent of supporting a two step commitment process.

Goldman at p. 5, lns. 10-14.

107. Further examples include Goldman at p. 12, lns. 10-16; p. 16, lns. 5- 25; Fig. 7.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

108. Goldman discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

The present patent document relates to a novel method for deployment of policy to a target connected to a network for the purpose of controlling the actions of that target based upon certain predefined conditions. In representative embodiments, methods are disclosed for creating another step in the policy deployment process within which ***policies can be created, tested, changed, and deleted*** prior to their transfer to the policy configuration agents of the targets to which it is intended that they will eventually be deployed.

Goldman at p. 4, lns. 3-9.

109. Further examples include Goldman at p. 2, ln 29 - p.3, ln. 3; p. 16, lns. 17-25; Fig. 7.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

110. Goldman discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

Operation 310 of figure 7, as in figure 3, assigns policy 120. The system 400 allows the user to assign the policy 120 to the target 110 on a per target 110 basis, i.e., given the target 110, present the list of possible policies 120 so that one can be assigned, or on a per policy 120 basis, i.e., given the policy 120, present the list of targets 110 which support the policy's 120 type so the policy 120 can be assigned to one of them. This operation will create and store the assignment relationship based on the target 110 as described above.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Operation 710 of figure 7 displays assigned policy 120 for a given target 110. The system 400 displays a list of targets 110. For each target 110, the system 400 displays its assigned policies 120 and committed policies 120. This requires the system 400 to support finding the assignment relationship for a given target 110 so the policy 120 can be displayed.

Operation 720 of figure 7 tests/simulates assigned policy 120 for a given target 110.

Operation 730 of figure 7 clears assigned policy 120 for a given target 110. The system 400 allows the user to clear the assigned policy 120 for a given target 110. In this case, the system clears the assigned policy 120 relationship for that target 110.

Operation 740 of figure 7 determines to which targets 110 the policy 120 is assigned. The system 400 allows the user to see to which targets 110 a given policy 120 is assigned. This operation is supported by a query which searches through the assignment relationships for entries which include a reference to the given policy 120.

Goldman at p. 16, lns. 5-25.

111. Further examples include Goldman at Fig. 7.

112. To the extent this limitation is not expressly disclosed by Goldman, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

**2. Claim 2**

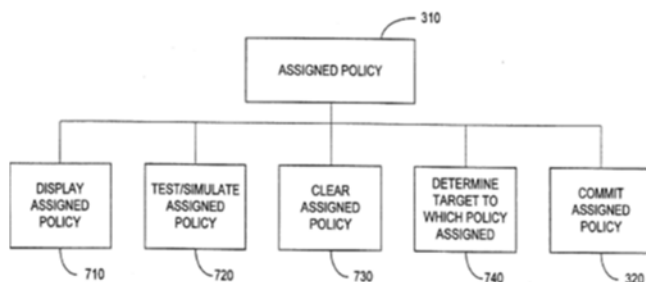
a. *The computer-implemented method of claim 1, wherein the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

113. Goldman discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example,



**FIG.7**

Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

114. To the extent this limitation is not expressly disclosed by Goldman, it would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency. Indeed, Goldman discloses improving efficiency in a distributed network with agents having one or more associated underlying resources.

### 3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

115. Goldman discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

The present patent document relates to a novel method for deployment of policy to a target connected to a network for the purpose of controlling the actions of that target ***based upon certain predefined conditions***. In representative embodiments, methods are disclosed for creating another step in the policy deployment process within which policies can be created, tested, changed, and deleted prior to their transfer to the policy

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

configuration agents of the targets to which it is intended that they will eventually be deployed.

Goldman at p. 4, lns. 3-9.

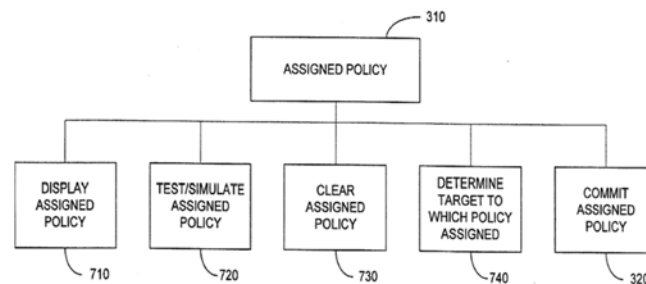
116. Further examples include Goldman at Claim 6.

117. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. **Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

118. Goldman discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example:



**FIG.7**

Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

119. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. **Claim 7**

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

120. Goldman discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

***A computer program storage medium [447] readable by a computer,*** tangibly embodying a computer program of instructions executable by the computer to perform method steps, the method steps comprising: assigning a policy [120] to a target [110], providing the policy [120] specifies conditional action implementable on the target [110], providing the target is a resource on a network [220], and providing policy [120] assignment comprises association of the policy [120] with the target [110] prior to policy [120] reconfiguration of the target [110]; and activating the policy [120] on the target [110], providing the policy [120] has been activated when target [110] actions comply with the policy [120].

Goldman at Claim 6.

121. Further examples include Goldman at p. 13, lns. 14-20; p. 13, lns. 21-29; Fig. 4A (showing a system for policy management by a server program for the target); Fig. 4B.

b. *running at least one thread in a first runtime environment;*

122. Goldman discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

123. Goldman discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

124. Goldman discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

125. Goldman discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

126. Goldman discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

127. Goldman discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**6. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

128. Goldman discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

**7. Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

129. Goldman discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

a. *A system, comprising:*

130. Goldman discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

The policy-based ***network management system*** supports a number of operations related to the two stage deployment mechanism comprising the following: (1) assignment of policy to targets on a per target basis which creates and stores the assignment relationship, (2) display of assigned policy, (3) tests and simulation of assigned policy, (4) clearing of assigned policy, (5) identify to which targets a given policy is assigned, and (6) commit an assigned policy to the target.

Goldman at p.6, lns. 4-9.

131. Further examples include Goldman at p. 13, lns. 14-20; p. 13, lns. 21-29; Figs. 4A-4B.

b. *a processor;*

132. Goldman discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example,

Figure 4A is a drawing of a system 400 for policy 120 management by a server program 410 for the target 110 as described in various representative embodiments of the present patent document. A console 430 connected to the server program 410 provides the user interface to enable the assignment of policy 120 to the appropriate targets 110 prior to commitment. The policy 120 is typically stored in a memory 445 located on ***a computer program storage medium 447*** connected to the server program 410, all of which could be located on a computer 405.

Goldman at p. 13, lns. 14-20.

133. Further examples include Goldman at p. 13, lns. 21-29; Fig. 4A (showing a system for policy management by a server program for the target); Fig. 4B.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*system to perform a method comprising:*

134. Goldman discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13.

d. *running at least one thread in a first runtime environment;*

135. Goldman discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

136. Goldman discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

137. Goldman discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

138. Goldman discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

139. Goldman discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

140. Goldman discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

141. Goldman discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**9. Claim 14**

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

142. Goldman discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**10. Claim 15**

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

143. Goldman discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

144. Goldman discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

145. I expect to testify that Goldman anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Goldman discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-3.

**C. Anticipation by and/or Obviousness in View of United States Patent No. 6,671,724, filed 3/21/2000 and issued to Pandya, et al. on 12/30/2003.**

146. As explained in detail below and in the chart attached as Ex. B-4, Pandya anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

147. Pandya discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

Software, systems and ***methods for managing a distributed network environment including a plurality of computers interconnected by a network link***, where at least some of the computers include a layered communications protocol stack for providing a data interface between an application program and the network link, the communications stack having a transport protocol layer for providing an end-to-end communications connection. The invention includes a control module and a plurality of agent modules, each agent being associated with one of the computers and adapted to dynamically monitor the associated computer at a data transmission point between an application program running on the computer and the transport protocol layer and repeatedly communicate with the control module in order to effect management of the distributed network system. The invented software, systems and methods may also include a messaging feature for providing users, IT personnel, or various management systems with informative messages concerning network conditions and network resources.

Pandya at Abstract.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

148. Pandya discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

Software, systems and methods for managing a distributed network environment including a plurality of computers interconnected by a network link, where at least some of the computers include a layered communications protocol stack for providing a data interface between an application program and the network link, the communications stack having a transport protocol layer for providing an end-to-end communications connection. The invention includes a control module and a ***plurality of agent modules, each agent being associated with one of the computers*** and adapted to dynamically monitor the associated computer at a data transmission point between an application program running on the computer and the transport protocol layer and repeatedly communicate with the control module in order to effect management of the distributed network system. The invented software, systems and methods may also include a messaging feature for providing users, IT personnel, or various management systems with informative messages concerning network conditions and network resources.

Pandya at Abstract.

149. Further examples include Pandya at 4:31-46, 7:27-39, 9:65-10:11, Figs. 4 and 6.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

150. Pandya discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

Referring now to FIGS. 6-9, the agent module will be more particularly described. ***The basic functions of the agent module are monitoring the status and activities of its associated client***, server, pervasive computing device or other computing device, communicating this information to one or more control points, ***enforcing system policies under the direction of the control points***, and providing messages to network users and administrators concerning network conditions. FIGS. 6-8 are conceptual depictions of networked computing devices, and show how the agent software is associated with the networked devices relative to layered protocol software used by the devices for network communication.

Pandya at 9:65-10:11.

151. Further examples include Pandya at 21:17-38, 8:7-9:31, Fig. 9.

d. *detecting if there is an abnormality in the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

152. Pandya discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

In addition, optimum and minimum performance levels may be established for applications or other tasks using network resources. Referring again to the IP telephony example discussed above, the configuration utility may be used to ***specify a minimum threshold performance level for a networked device running the IP telephony application.*** This performance level may be specified in terms of QoS performance parameters such as bandwidth, throughput, jitter, delay and loss. ***The agent module associated with the networked device would then monitor the network traffic associated with the IP telephony application to ensure that performance was above the minimum threshold. If the minimum level was not met, the control points and agents could interact to reallocate resources and provide the specified minimum service level.*** Similarly, an optimum service level may be specified for various network applications and tasks. More generally, configuration utility 106 may be configured to manage system policies by providing functionality for authoring, maintaining and storing system policies, and for managing retrieval of system policies from other locations on a distributed network, such as a dedicated policy server.

Pandya at 21:17-38.

153. Further examples include Pandya at 8:7-9:31, 18:48-19:44, Fig. 12.

e. *and performing a corrective action to fix any detected abnormalities,*

154. Pandya discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1.

For example, successful businesses often strive to provide excellent customer services. ***This underlying business goal can be translated into many different policies defining how network resources are to be used. One example of such a policy would be to prevent or limit access to non-business critical applications when performance of business critical applications is degraded beyond a threshold point. Another example would be to use QoS techniques to provide a guaranteed or high level of service to e-commerce applications. Yet another example would be to dynamically increase the network bandwidth allocated to a networked computer whenever it is accessed by a customer. Also, bandwidth for various applications might be restricted during times when there is heavy use of network resources by customers.***

***Control points 72 would access these policies and provide policy data to agents 70.*** Agents 70 and control points 72 would communicate with each other and monitor the network to determine how many customers were accessing the network, what computers the customer(s) were accessing, and what applications were being accessed by the customers. Once the triggering conditions were detected, the agents and control points would

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

interact to re-allocate bandwidth, provide specified service levels, block or restrict various non-customer activities, etc.

*Another example of policy-based management would be to define an optimum specification of network resources or service levels for particular types of network tasks.* The particular policies would direct the management entities to determine whether the particular task was permitted, and if permitted, the management entities would interact to ensure that the desired level of resources was provided to accomplish the task. If the optimum resources were not available, the applicable policies could further specify that the requested task be blocked, and that the requesting user be provided with an informative message detailing the reason why the request was denied. Alternatively, the policies could specify that the user be provided with various options, such as proceeding with the requested task, but with sub-optimal resources, or waiting to perform the task until a later time.

For example, continuous media applications such as IP telephony have certain bandwidth requirements for optimum performance, and are particularly sensitive to network jitter and delay. *Policies could be written to specify a desired level of service, including bandwidth requirements and threshold levels for jitter and delay, for client computers attempting to run IP telephony applications. The policies would further direct the agents and control modules to attempt to provide the specified level of service.* Security checking could also be included to ensure that the particular user or client computer was permitted to run the application. In the event that the specified service level could not be provided, the requesting user could be provided with a message indicating that the resources for the request were not available. The user could also be offered various options, including proceeding with a sub-optimal level of service, placing a conventional telephone call, waiting to perform the task until a later time, etc.

The software, system and methods of the present invention may be used to implement a wide variety of system policies. *The policy rules and conditions may be based on any number of parameters, including IP source address, IP destination address, source port, destination port, protocol, application identity, user identity, device identity, URL, available device bandwidth, application profile, server profile, gateway identity, router identity, time-of-day, network congestion, network load, network population, available domain bandwidth and resource status,* to name but a partial list. *The actions taken when the policy conditions are satisfied can include blocking network access, adjusting service levels and/or bandwidth allocations for networked devices, blocking requests to particular URLs, diverting network requests away from overloaded or underperforming resources, redirecting network requests to alternate resources and gathering network statistics.*

*Some of the parameters listed above may be thought of as "client parameters," because they are normally evaluated by an agent monitoring a single networked client device. These include IP source address, IP destination address, source port, destination port, protocol, application identity, user identity, available device bandwidth and URL. Other parameters, such as application profile, server profile, gateway identity,*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*router identity, time-of-day, network congestion, network load, network population, available domain bandwidth and resource status may be thought of as “system parameters” because they pertain to shared resources, aggregate network conditions or require evaluation of data from multiple agent modules.* Despite this, there is not a precise distinction between client parameters and system parameters. Certain parameters, such as time-of-day, may be considered either a client parameter or a system parameter, or both.

Pandya at 8:7-9:31.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

155. Pandya discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example, Pandya discloses making a request to a control point for a different bandwidth allocation when it cannot meet the quality thresholds established in its service policies. This request can be made to a control point, for example, to increase its bandwidth allocation, when it cannot create a corrective policy to lower other agents’ bandwidth. *See, e.g.,* Pandya at 8:7-9:31.

156. Further examples include Pandya at 4:31-46, 16:28-17:35, 17:36-18:44, 18:48-19:44, 21:17-38, Figs. 11A-D and 12.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

157. Pandya discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

FIGS. 11A, 11B, 11C and 11D depict examples of various methods that may be implemented by traffic module 160 to dynamically allocate bandwidth. FIG. 11A depicts a process by which traffic module 160 determines whether any adjustments to bandwidth allocations AB are necessary.... Second, if there are any agents for which  $AB < CB$  and  $UB \cong AB$ , the allocation for those agents is modified, as seen in steps S6 and S10. The allocations for any such agent are typically increased.... Third, if

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

there are any agents reporting bandwidth usage UB that is less than their allocation AB, as determined at step S8, then the allocation AB for such an agent is reduced for the upcoming period to free up the unused bandwidth. Steps S4, S6 and S8 may be performed in any suitable order. Collectively, these three steps ensure that certain bandwidth allocations are modified, i.e. increased or reduced, if one or more of the following three conditions are true: (1)  $AB_{total} > CB_{total}$ , (2)  $AB < CB$  and  $UB \cong AB$  for any agent, or (3)  $UB < AB$  for any agent. If none of these are true, the allocations AB from the prior period are not adjusted. Traffic module 160 modifies allocations AB as necessary at step S10. After all necessary modifications are made, the control point communicates the new allocations to the agents for enforcement during the upcoming cycle.

Pandya at 16:28-17:35.

158. Further examples include Pandya at 17:36-18:44, 18:48-19:44, Figs. 11A-D.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

159. Pandya discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

The invention includes two main software components, an agent and a control module, also referred to as a control point. The agents and control points are deployed throughout distributed network 10, and interact with each other to accomplish the above goals. A plurality of agents may be deployed to intelligently couple clients, servers and other computing devices to the underlying network. The deployed agents monitor, analyze and act upon network events relating to the networked devices with which they are associated. The agents are centrally coordinated and/or controlled by one or more control points. The agents and control points interact to control and monitor network events, track operational and congestion status of network resources, select optimum targets for network requests, dynamically manage bandwidth usage, and share information about network conditions with customers, users and IT personnel.

Pandya at 4:31-46.

160. Further examples include Pandya at 7:27-39, 9:65-10:11, 14:35-44.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

161. Pandya discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example, Pandya’s bandwidth allocation mechanism comprises load balancing. Further examples include.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

162. Pandya discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

In addition, optimum and minimum performance levels may be established for applications or other tasks using network resources. Referring again to the IP telephony example discussed above, the configuration utility may be used to ***specify a minimum threshold performance level for a networked device running the IP telephony application.*** This performance level may be specified in terms of QoS performance parameters such as bandwidth, throughput, jitter, delay and loss. ***The agent module associated with the networked device would then monitor the network traffic associated with the IP telephony application to ensure that performance was above the minimum threshold. If the minimum level was not met, the control points and agents could interact to reallocate resources and provide the specified minimum service level.*** Similarly, an optimum service level may be specified for various network applications and tasks. More generally, configuration utility 106 may be configured to manage system policies by providing functionality for authoring, maintaining and storing system policies, and for managing retrieval of system policies from other locations on a distributed network, such as a dedicated policy server.

Pandya at 21:17-38.

163. Further examples include Pandya at 8:7-9:31, 18:48-19:44.

164. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Algorithm.*

165. Pandya discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6.

166. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

167. Pandya discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

Software, systems and ***methods for managing a distributed network environment including a plurality of computers interconnected by a network link***, where at least some of the computers include a layered communications protocol stack for providing a data interface between an application program and the network link, the communications stack having a transport protocol layer for providing an end-to-end communications connection. The invention includes a control module and a plurality of agent modules, each agent being associated with one of the computers and adapted to dynamically monitor the associated computer at a data transmission point between an application program running on the computer and the transport protocol layer and repeatedly communicate with the control module in order to effect management of the distributed network system. The invented software, systems and methods may also include a messaging feature for providing users, IT personnel, or various management systems with informative messages concerning network conditions and network resources.

Pandya at Abstract.

168. Further examples include Pandya at 4:31-46, 6:42-59.

169. To the extent this limitation is not expressly disclosed by Pandya, it would have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in a distributed networking system at least to improve system efficiency.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *running at least one thread in a first runtime environment;*

170. Pandya discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

171. Pandya discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

172. Pandya discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- e. *and performing a corrective action to fix any detected abnormalities;*

173. Pandya discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

174. Pandya discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

175. Pandya discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

176. Pandya discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

177. Pandya discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

178. Pandya discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

Software, systems and ***methods for managing a distributed network environment including a plurality of computers interconnected by a network link***, where at least some of the computers include a layered communications protocol stack for providing a data interface between an application program and the network link, the communications stack having a transport protocol layer for providing an end-to-end communications connection. The invention includes a control module and a plurality of agent modules, each agent being associated with one of the computers and adapted to dynamically monitor the associated computer at a data transmission point between an application program running on the computer and the transport protocol layer and repeatedly communicate with

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the control module in order to effect management of the distributed network system. The invented software, systems and methods may also include a messaging feature for providing users, IT personnel, or various management systems with informative messages concerning network conditions and network resources.

Pandya at Abstract.

b. *a processor;*

179. Pandya discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

***The agents monitor network resources and the activity of the device with which they are associated***, and communicate this information to the control points. In response to monitored network conditions and data reported by agents, the control points alter the behavior of particular agents in order to provide the desired network services. ***The control points and agents may be loaded on a wide variety of devices, including general purpose computers, servers, routers, hubs, palm computers, pagers, cellular telephones, and virtually any other networked device having a processor and memory.*** Agents and control points may reside on separate devices, or simultaneously on the same device.

Pandya at 7:27-39.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

180. Pandya discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example,

***The agents monitor network resources and the activity of the device with which they are associated***, and communicate this information to the control points. In response to monitored network conditions and data reported by agents, the control points alter the behavior of particular agents in order to provide the desired network services. ***The control points and agents may be loaded on a wide variety of devices, including general purpose computers, servers, routers, hubs, palm computers, pagers, cellular telephones, and virtually any other networked device having a processor and memory.*** Agents and control points may reside on separate devices, or simultaneously on the same device.

Pandya at 7:27-39.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *running at least one thread in a first runtime environment;*

181. Pandya discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

182. Pandya discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

183. Pandya discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

184. Pandya discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

185. Pandya discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

186. Pandya discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

187. Pandya discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

188. Pandya discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

189. Pandya discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises using Dijkstra's Self Stabilization Algorithm.*

190. Pandya discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

191. I expect to testify that Pandya anticipates and/or renders obvious each Asserted Claim of the '659 Patent. A claim chart illustrating that Pandya discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-4.

**D. Anticipation by and/or Obviousness in View of United States Patent No. 6,012,152, issued to Douik, et al. on 1/4/2000.**

192. As explained in detail below and in the chart attached as Ex. B-5, Douik anticipates and renders obvious the claims of the '659 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

a. *A computer-implemented method, comprising:*

193. Douik discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” '659 Pat. at Cl. 1. For example:

***Network management*** means deploying and coordinating resources in order to plan, operate, administer, analyze, evaluate, design and expand communication networks to meet service level objectives at all times, at reasonable cost and with optimum capacity. Network management developments for mobile networks have almost the same objectives as for wired networks, the main objectives being to ensure good operation and service provisioning.

Douik at 2:41-48.

194. Further examples include Douik at 8:60-9:3, 32:5-18, Fig. 4.

b. *running at least one thread in a first runtime environment;*

195. Douik discloses this element of claim 1, “running at least one thread in a first runtime environment.” '659 Pat. at Cl. 1. For example:

Management information for telecommunication systems is not always found at one single—logical or physical—location. Normally, the management of a large network is distributed over various managers which manage (arbitrary) parts of the network. ***This means that the model of the overall network is cut into pieces and stored at different managers. In the***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*area of [Software Fault Management (SFM)], several SFM Systems may cooperate, with each one being responsible for a different part of the overall model.* Whenever managers need information beyond their model knowledge, they ask higher level managers which in turn have the right to request information from all subordinate managers. With the cooperation and the necessary interfaces between the model parts, boundaries between management domains....

Douik at 25:27-41.

196. Further examples include Douik at Fig. 1.

197. To the extent this limitation is not expressly disclosed by Douik, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

198. Douik discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. ***The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.***

Douik at 13:15-22.

199. Further examples include Douik at 32:49-33:3.

200. To the extent this limitation is not expressly disclosed by Douik, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

d. *detecting if there is an abnormality in the monitored operational parameters;*

201. Douik discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. ***The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.***

Douik at 13:15-22.

202. Further examples include Douik at 32:49-33:3.

e. *and performing a corrective action to fix any detected abnormalities,*

203. Douik discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example,

At step 172, it is determined whether the suspected fault is a known fault. ***If the fault is already known, the process moves to step 172 and implements and deploys corresponding corrective actions. However, if the fault is not a known fault, the process moves to step 174 where further fault analysis is performed.*** Once again, human expert help from step 168 may be utilized as well as additional network monitoring data at 175. Following this analysis, diagnostic tests are performed at 176, and the SFM system may interface various test tools at 177 for this purpose.

Douik at 36:62-37:4.

204. Further examples include Douik at 38:13-23; Douik at Figs. 5, 6.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

205. Douik discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Management information for telecommunication systems is not always found at one single-logical or physical location. Normally, the management of a large network is distributed over various managers which manage (arbitrary) parts of the network. This means that the model of the overall network is cut into pieces and stored at different managers. In the area of [Software Fault Management (SFM)], several SFM Systems may cooperate, with each one being responsible for a different part of the overall model. ***Whenever managers need information beyond their model knowledge,***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*they ask higher level managers which in turn have the right to request information from all subordinate managers.* With the cooperation and the necessary interfaces between the model parts, boundaries between management domains....

Douik at 25:27-41.

206. Further examples include Douik at Fig. 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

207. Douik discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example, if an agent does not have enough information to diagnose a fault, it can ask a neighboring agent for more information (a corrective policy). If the new information results in a diagnosis of the fault, the agent can then repair the fault.

At step 172, it is determined whether the suspected fault is a known fault. If the fault is already known, the process moves to step 172 and implements and deploys corresponding corrective actions. However, if the fault is not a known fault, the process moves to step 174 where further fault analysis is performed. Once again, human expert help from step 168 may be utilized as well as additional network monitoring data at 175. Following this analysis, diagnostic tests are performed at 176, and the SFM system may interface various test tools at 177 for this purpose.

Douik at 36:62-37:4.

208. Further examples include Douik at 38:13-23; Douik at Figs. 5, 6.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

209. Douik discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. *The SFM agents, working on different network elements and/or on different aspects of the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.*

Douik at 13:15-22.

210. Further examples include Douik at 37:25-42.

211. To the extent this limitation is not expressly disclosed by Douik, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

**2. Claim 2**

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

212. Douik discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. *The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.*

Douik at 13:15-22.

213. Further examples include Douik at 32:49-33:3.

214. To the extent this limitation is not expressly disclosed by Douik, it would have been obvious to a person of ordinary skill in the art to perform corrective actions that included load balancing operations in a distributed networking system at least to improve network efficiency and scalability.

**3. Claim 3**

a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters to known thresholds.*

215. Douik discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example,

*The Case Based Reasoning approach uses a knowledge base built of standard cases.* Each case has to be coded as scripts based on the experience gained from the working system. The different cases represent a well-defined application field. Each problem handled by the reasoning mechanism is, if possible, mapped into an existing case stored in the knowledge base. Hence, this technique is suitable for applications, which can be reduced to a small set of already available and known cases. This means that the development of the case base has to be completed in order for the case knowledge to be available.

Douik at 20:22-32.

216. Further examples include Douik at 22:49-61.

217. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

**4. Claim 6**

a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

218. Douik discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example:

***Automated Fault Management***

There are several existing knowledge-based and artificial intelligence (AI) techniques that can be used for fault diagnosis. Five categories relevant to fault diagnosis are identified: fault-based techniques, model-based techniques, case-based reasoning techniques, machine learning for knowledge acquisition, and integrated diagnostic techniques. A description of the techniques and how they apply to diagnosis follows.

Douik at 6:31-39.

219. Further examples include Douik at 31:23-37.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

220. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

221. Douik discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. For example:

***Network management*** means deploying and coordinating resources in order to plan, operate, administer, analyze, evaluate, design and expand communication networks to meet service level objectives at all times, at reasonable cost and with optimum capacity. Network management developments for mobile networks have almost the same objectives as for wired networks, the main objectives being to ensure good operation and service provisioning.

Douik at 2:41-48.

222. Further examples include Douik at 8:60-9:3, 32:5-18; Douik at Fig. 4.

- b. *running at least one thread in a first runtime environment;*

223. Douik discloses this element of claim 7, "running at least one thread in a first runtime environment," at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

224. Douik discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

225. Douik discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *and performing a corrective action to fix any detected abnormalities;*

226. Douik discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

227. Douik discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

228. Douik discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

229. Douik discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters to known thresholds.*

230. Douik discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

a. *A system, comprising:*

231. Douik discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

***Network management*** means deploying and coordinating resources in order to plan, operate, administer, analyze, evaluate, design and expand communication networks to meet service level objectives at all times, at reasonable cost and with optimum capacity. Network management developments for mobile networks have almost the same objectives as for wired networks, the main objectives being to ensure good operation and service provisioning.

Douik at 2:41-48.

232. Further examples include Douik at 8:60-9:3, 32:5-18; Douik at Fig. 4.

b. *a processor;*

233. Douik discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

***Network management*** means deploying and coordinating resources in order to plan, operate, administer, analyze, evaluate, design and expand communication networks to meet service level objectives at all times, at reasonable cost and with optimum capacity. Network management developments for mobile networks have almost the same objectives as for wired networks, the main objectives being to ensure good operation and service provisioning.

Douik at 2:41-48

234. Further examples include Douik at 8:60-9:3, 32:5-18; Douik at Fig. 4.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*system to perform a method comprising:*

235. Douik discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

***Network management*** means deploying and coordinating resources in order to plan, operate, administer, analyze, evaluate, design and expand communication networks to meet service level objectives at all times, at reasonable cost and with optimum capacity. Network management developments for mobile networks have almost the same objectives as for wired networks, the main objectives being to ensure good operation and service provisioning.

Douik at 2:41-48.

236. Further examples include Douik at 8:60-9:3, 32:5-18; Douik at Fig. 4.

d. *running at least one thread in a first runtime environment;*

237. Douik discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

238. Douik discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

239. Douik discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

240. Douik discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

241. Douik discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

242. Douik discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

243. Douik discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

244. Douik discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

245. Douik discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

246. Douik discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

247. I expect to testify that Douik anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Douik discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-5.

**E. Anticipation by and/or Obviousness in View of United States Patent No. 6,584,502, filed 6/29/1999 and issued to Natarajan, et al. on 6/24/2003.**

248. As explained in detail below and in the chart attached as Ex. B-7, Natarajan anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

249. Natarajan discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

As the use of computer networks proliferates, there exists an increasing need to improve computer network designs and implementations in order to facilitate the *management, implementation, and modification of such networks*.

Natarajan at 2:11-15.

250. Further examples include Natarajan at 2:17-24, 2:25-28, 6:7-7:11, 7:54-64.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

251. Natarajan discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

In a specific embodiment of this invention, the technique of the present invention is implemented in software such as an *operating system or in an application running on an operating system.*

Natarajan at 11:31-34.

252. Further examples include Natarajan at 11:57-12:17, 5:7-37, 6:23-32, 27:4-11, 14:8-12, 14:20-27, 26:60-62, 15:46-55, 20:48-60, 25:38-55, 27:27-41, 28:41-48, 22:36-44, 32:45-46; Natarajan at Figs. 2, 16.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

253. Natarajan discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

Typically, when a client is provisioned a virtual circuit, a service provider will guarantee to provide a minimum amount of bandwidth for handling client traffic. More specifically, the consumer pays for, and the service provider agrees to provide, a virtual circuit which is able to handle a specified *committed burst size* ( $B_c$ ) of data (e.g.  $B_c=256K$  bits) over a specified committed time interval ( $T_c$ ) (e.g.  $T_c=1$  second). The value  $B_c/T_c$  is referred to as the *committed information rate* (CIR), which specifies the rate at which the virtual circuit will accept data from the consumer. In the example above, the CIR value is equal to 256K bits/sec. meaning that over the span of one second, the service provider guarantees to handle 256 Kbits of data.

At times, however, *a client may be running an application* which requires more than 256K bits/sec of bandwidth. For example, the client may be required to send bursts of traffic at 300K bits/sec from time to time. Although the CIR value represents the average guaranteed bandwidth for the virtual circuit, the service provider also provides an allowance for data bursts. This allowance is referred to as the *excess burst size* ( $B_e$ ) (e.g.  $B_e=64K$ bits). Thus, using the example above, in time  $T_c$  (1 second), the  $B_c$  value (256K) may be exceeded by  $B_e$  (64K) without the customer having to pay. This amount of additional bandwidth is referred to as the *excess information rate* ( $EIR=B_e/T_c$ ), and represents the excess bandwidth which the service provider will attempt to deliver to the consumer. *Any data sent by the consumer in excess of the EIR value will be dropped. Further, data bursts which are between the  $B_c$  and  $B_e$  value may be discarded in the event of network congestion.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Natarajan at 5:7-37.

254. Further examples include Natarajan at 6:23-32, 6:66-7:11, 7:19-24, 20:20-27, 20:41-44, 20:48-60, 25:38-55, 15:5-18, 15:19-29, 27:27-41, 27:54-28:1, 28:15-19, 29:30-36, 30:17-22, 33:49-52, 15:46-55, 32:45-46, 28:41-48, 33:16-20, 17:62-18:5, Fig. 17.

255. To the extent this limitation is not expressly disclosed by Natarajan, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

d. *detecting if there is an abnormality in the monitored operational parameters;*

256. Natarajan discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

At 1208, the dropped packet count values from links B and C are each compared with the threshold dropped packet count value (set during initialization at 1202). If the dropped packet count value for either of links B or C do not exceed the established threshold value, then it may be inferred that the current CIR values for links B and C are adequate for achieving a desired network performance which conforms with predetermined criteria or specified guidelines. In the example of FIG. 12, network performance is determined using predetermined criteria which is related to the threshold value of the dropped packet count for links B and C. Thus, if the dropped packet counts for links B and C conform with (i.e., are less than or equal) the predetermined criteria specified by the threshold dropped packet count value, that particular aspect of network performance is determined to be adequate. Accordingly, the existing CIR values for links B and C need not be modified or updated at the present time. The CIR analysis then continues at block 1218 wherein the frame relay CIR policy procedure waits (1218) a predetermined time interval (T) (e.g. 5 sec–30 sec.) before either repeating the procedure (1220) or exiting the procedure (1224).

If either of the dropped packet count values from links B or C exceeds the threshold dropped packet count value, it may be inferred that at least one of the CIR values for links B and/or C is inadequate for maintaining a dropped packet count for each of the links below the threshold value. Accordingly, in order to adapt to this change in the network conditions, the CIR values for links B and C are recomputed.

Natarajan at 17:12-40.

257. Further examples include Natarajan at 28:3-24, 31:35-32:22.

e. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities,*

258. Natarajan discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

***The updated CIR data may then be used by various elements in the network to modify or affect each element's respective behavior or operation*** (thereby affecting operation of the network). In this way, the network elements are able to automatically and dynamically adapt to changing network conditions.

Natarajan at 15:24-29

259. Further examples include Natarajan at 16:26-45, 18:19-38, 18:42-59, 20:28-40, 24:54-62, 28:46-48, 28:59-29:3, 29:33-36; Natarajan at Figs. 10, 17, 18.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

260. Natarajan discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

An additional aspect of this embodiment provides that, where the condition or status of the first device relates to an error detected by the first device, an event notification message relating to the error is reported to at least one other network device.

Natarajan at 3:17-21.

261. Further examples include Natarajan at 7:12-19, 7:24-32, 10:31-57, 11:31-34, 14:2-4, 14:15-19, 14:40-56, 15:8-11, 15:12-18, 18:42-59, 19:25-35, 21:12-16, 23:62-24:9, 29:43-54, 32:39-46, 32:17-20, 33:9-53, 28:11-21, 15:12-18, 14:56-67, 32:27-38, 33:9-53, 27:4-11, 30:36-41, 30:35-45, 26:60-62, 16:59-17:2; Natarajan at Figs. 2, 9B, 13, 16.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

262. Natarajan discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

The definition of “policy” varies based upon the perspective of the user. In the case of management and control of network elements, **a policy is a corrective action** which is used to restore the network element to a pre-determined state.

Natarajan at 14:15-19.

263. Further examples include Natarajan at 15:8-11, 14:40-56, 7:24-32, 32:39-46, 15:12-18, 14:56-67, 32:27-38, 33:9-53.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

264. Natarajan discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

Referring to FIG.16, a frame relay virtual circuit is shown connecting user1 (1602) with user2 (1620). User1 communicates with a first router 1604 via **link A**. In the example of FIG. 16, router 1604 may be managed by a first service provider (SP1). Router 1604 communicates with a frame relay cloud 1612 via **link B** and Switch 1612. The frame relay cloud 1610 is managed and maintained by a service provider such as AT&T. The frame relay cloud 1610 communicates with a second router 1614 via **link C** and Switch 1613. The second router 1614 may be managed by a second service provider (SP2). user 1620 communicates with router 1614 via **link D**.

Natarajan at 29:43-54.

265. Further examples include Natarajan at 18:42-59, 14:40-56, 32:39-46, 19:25-35, 11:31-34, 20:23-27, 13:63-14:2, 17:3-5, 19:53-57, 3:22-42, 3:13-17, 8:10-13, 8:21-28, 2:32-34, 30:46-49, 32:39-46, 7:12-19, 8:39-47, 26:67-27:3, 14:56-67, 26:50-27:11, 30:35-45, 14:20-27, 16:59-17:2, 27:12-26, 30:35-45, 31:35-47, 10:41-57, 33:5-12, 2:53-56; Natarajan at Figs. 2, 3, 4, 8, 16.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

266. Natarajan discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

Using the technique of the present invention, a dynamic feedback-based adaptive network may be provided for automatically ***detecting a client’s need for increased bandwidth***, and for automatically and dynamically reconfiguring the appropriate network elements to ***provide sufficient bandwidth*** on the virtual circuit to support the user’s current application(s). The feedback-based adaptive network of the present invention monitors current conditions of local and/or remote network elements and dynamically adjusts network control parameters based upon analysis of the monitored network elements. A specific embodiment of the dynamically configurable feedback-based adaptive network of the present invention is shown in FIG. 2 of the drawings.

Natarajan at 6:66-7:11.

267. Further examples include Natarajan at 8:39-51, 14:52-67, 15:19-29, 16:46-18:38, 20:23-27, 25:44-55, 26:24-33, 27:54-59; Natarajan at Figs. 12, 17, 18.

**3. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

268. Natarajan discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

At 1208, ***the dropped packet count values from links B and C are each compared with the threshold dropped packet count value (set during initialization at 1202)***. If the dropped packet count value for either of links B or C do not exceed the established threshold value, then it may be inferred that the current CIR values for links B and C are adequate for achieving a desired network performance which conforms with predetermined criteria or specified guidelines. In the example of FIG. 12, network performance is determined using predetermined criteria which is related to the threshold value of the dropped packet count for links B and C. Thus, if the dropped packet counts for links B and C conform with (i.e., are less than or equal) the predetermined criteria Specified by the threshold dropped packet count value, that particular aspect of network performance is determined to be adequate. Accordingly, the existing CIR values for links B and C need not

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

be modified or updated at the present time. The CIR analysis then continues at block 1218 wherein the frame relay CIR policy procedure waits (1218) a predetermined time interval (T) (e.g. 5 sec–30 sec.) before either repeating the procedure (1220) or exiting the procedure (1224).

*If either of the dropped packet count values from links B or C exceeds the threshold dropped packet count value, it may be inferred that at least one of the CIR values for links B and/or C is inadequate for maintaining a dropped packet count for each of the links below the threshold value.* Accordingly, in order to adapt to this change in the network conditions, the CIR values for links B and C are recomputed.

Natarajan at 17:12-40.

269. Further examples include Natarajan at 28:3-24, 31:35-32:22.

270. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

271. Natarajan discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example, Natarajan discloses that "the policy engine may be implemented with *neural networks* or with *other artificial intelligence technologies*." Natarajan at 14:30-32. Further examples include Natarajan at 28:59-29:3, 7:54-64.

272. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

273. Natarajan discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

Because such information and program instructions may be employed to implement the systems/methods described herein, ***the present invention relates to machine readable media*** that include program instructions, operating information, etc. for performing various operations described herein. Examples of machine-readable media include, but are not limited to, magnetic media such as hard disks, floppy disks, and magnetic tape, optical media such as CD-ROM disks; magneto-optical media such as floptical disks, and hardware devices that are specially configured to store and perform program instructions, such as read-only memory devices (ROM) and random access memory (RAM). The invention may also be embodied in a carrier wave travelling over an appropriate medium such as airwaves, optical lines, electric lines, etc. Examples of program instructions include both machine code, such as produced by a compiler, and files containing higher level code that may be executed by the computer using an interpreter.

Natarajan at 12:60-13:11

274. Further examples include Natarajan at 11:25-31, 14:20-29, 8:13-20.

b. *running at least one thread in a first runtime environment;*

275. Natarajan discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

276. Natarajan discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

277. Natarajan discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

278. Natarajan discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

279. Natarajan discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

280. Natarajan discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**6. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

281. Natarajan discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

**7. Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

282. Natarajan discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

a. *A system, comprising:*

283. Natarajan discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

In a specific embodiment of this invention, the technique of the present invention is implemented in software such as an ***operating system or in an application running on an operating system.***

Natarajan at 11:31-34.

284. Further examples include Natarajan at 11:57-12:17, 14:8-12, 14:20-27, 26:61-62, 27:4-11.

b. *a processor;*

285. Natarajan discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

In the network of FIG. 2, network element 204A may be any ***hardware*** or software component which has a measurable parameter that can be reported. Examples of network elements include routers, switches, hosts, modems, terminals, dial access servers, gateways, ports, channels, interfaces, circuits, processes, drivers, protocols, services, applications, etc.

Natarajan at 8:13-20.

286. Further examples include Natarajan at 11:35-12:38, 21:28-35, 11:25-31, 14:8-12, 14:20-27, 27:4-11.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

287. Natarajan discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

According to the embodiment of FIG. 2, *each network element* which reports its respective operating information to data store 252, such as, for example network element 204A, *includes a local cache* 276A and an event handler (EH) 274A for handling event notification/registration. *Cache 276A represents any type of memory device which may be used for storing and/or caching updated control information received at the network element.* In a specific embodiment, this memory device may be a persistent store. After the updated control information has been cached into local cache 276A, the network element retrieves the updated control information from the cache and re-configures itself using this updated control information. Additionally, as shown in FIG. 2, network elements 208A and 208B may also include respective local caches (C) 276A, 276B for storing updated control information, and may also include respective event handling entities (EH) for receiving/reporting event notification information.

Natarajan at 8:52-9:2.

288. Further examples include Natarajan at 9:40-44, 9:55-10:2, 12:46-59, 14:8-12, 14:20-27, 27:4-11.

d. *running at least one thread in a first runtime environment;*

289. Natarajan discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

290. Natarajan discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

291. Natarajan discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

292. Natarajan discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

293. Natarajan discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

294. Natarajan discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

295. Natarajan discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

296. Natarajan discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

297. Natarajan discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

298. Natarajan discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

299. I expect to testify that Natarajan anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Natarajan discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-7.

**F. Anticipation by and/or Obviousness in View of United States Patent No. 5,655,081, issued to Bonnell, et al. on 8/5/1997.**

300. As explained in detail below and in the chart attached as Ex. B-15, Bonnell anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

301. Bonnell discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

It is another object of the present invention to provide an agent system for use in an enterprise management system wherein ***the agent system utilizes the memory and CPU resources of a server computer system in an efficient manner***, regardless of the number of console systems that are monitoring the resources on the server.

Bonnell at 6:28-33

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

302. Further examples include Bonnell at 6:61-7:8; Bonnell at Fig. 11.

b. *running at least one thread in a first runtime environment;*

303. Bonnell discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. ***An agent software system is installed on and runs on each of the server computer systems in the network.*** Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

304. Further examples include Bonnell at Fig. 11.

305. To the extent this limitation is not expressly disclosed by Bonnell, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

306. Bonnell discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is installed on and runs on each of the server computer systems in the network. ***Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bonnell at 6:61-7:8.

307. Further examples include Bonnell at Figs. 5b, 9, 11.

308. To the extent this limitation is not expressly disclosed by Bonnell, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

d. *detecting if there is an abnormality in the monitored operational parameters;*

309. Bonnell discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is installed on and runs on each of the server computer systems in the network. ***Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.***

Bonnell at 6:61-7:8.

310. Further examples include Bonnell at Figs. 9, 11.

e. *and performing a corrective action to fix any detected abnormalities,*

311. Bonnell discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is installed on and runs on each of the server computer systems in the network. ***Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system,***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*and executing recovery actions automatically when such actions are warranted.*

Bonnell at 6:61-7:8.

312. Further examples include Bonnell at Figs. 9, 11.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

313. Bonnell discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example, Bonnell at Fig. 8 illustrates this element.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

314. Bonnell discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. ***An agent software system is installed on and runs on each of the server computer systems in the network. Each respective agent software system carries out tasks on the computer system in which it is installed*** such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

315. Further examples include Bonnell at Figs. 8, 11.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

316. Bonnell discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. ***The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network.*** An agent software system is installed on and runs on each of the server computer systems in the network. Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:64-7:8.

317. Further examples include Bonnell at Figs. 11, 13.

318. To the extent this limitation is not expressly disclosed by Bonnell, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

2. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

319. Bonnell discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example, Bonnell at Fig. 5b illustrates this element.

320. To the extent this limitation is not expressly disclosed by Bonnell, it would have been obvious to a person of ordinary skill in the art to perform corrective actions that included load



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

balancing operations in a distributed networking system at least to improve network efficiency and scalability.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

321. Bonnell discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

In step 340, the agent sends the calculated parameter values to all consoles registered to receive “real-time” data for that instance and parameter. ***In step 342, the agent compares the value of every parameter against a threshold (usually stored in a knowledge module) to determine if a threshold-crossing event has occurred.*** If so, then the agent continues with step 344, in which it executes the event processing routing of FIG. 25. If not, the agent arrives at step 346 and repeats the loop by returning to step 330 if more resources are to be monitored.

Bonnell at 13:52-61.

322. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

323. Bonnell discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. ***An agent software system is installed on and runs on each of the server computer systems in the network.*** Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

324. Further examples include Bonnell at Fig. 11.

325. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

326. Bonnell discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

It is another object of the present invention to provide an agent system for use in an enterprise management system wherein ***the agent system utilizes the memory and CPU resources of a server computer system in an efficient manner***, regardless of the number of console systems that are monitoring the resources on the server.

Bonnell at 6:28-33.

327. Further examples include Bonnell at 6:61-7:8, Fig. 11.

- b. *running at least one thread in a first runtime environment;*

328. Bonnell discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

329. Bonnell discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters;and*

330. Bonnell discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *performing a corrective action to fix any detected abnormalities;*

331. Bonnell discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

332. Bonnell discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

333. Bonnell discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

a. *The computer-readable medium of claim 7, wherein the corrective*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

334. Bonnell discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

335. Bonnell discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

336. Bonnell discloses the preamble of claim 13, “[a] system, comprising . . .” ’659 Pat. at Cl. 13. For example:

It is another object of the present invention to provide an agent system for use in an enterprise management system wherein ***the agent system utilizes the memory and CPU resources of a server computer system in an efficient manner***, regardless of the number of console systems that are monitoring the resources on the server.

Bonnell at 6:28-33.

337. Further examples include Bonnell at 6:61-7:8, Fig. 11.

- b. *a processor;*

338. Bonnell discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

It is another object of the present invention to provide an agent system for use in an enterprise management system wherein ***the agent system utilizes the memory and CPU resources of a server computer system in an efficient manner***, regardless of the number of console systems that are monitoring the resources on the server.

Bonnell at 6:28-33.

339. Further examples include Bonnell at 6:61-7:8, Fig. 11.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

340. Bonnell discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

It is another object of the present invention to provide an agent system for use in an enterprise management system wherein ***the agent system utilizes the memory and CPU resources of a server computer system in an efficient manner***, regardless of the number of console systems that are monitoring the resources on the server.

Bonnell at 6:28-33.

341. Further examples include Bonnell at 6:61-7:8, Fig. 11.

- d. *running at least one thread in a first runtime environment;*

342. Bonnell discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

343. Bonnell discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

- f. *detecting there is an abnormality in the monitored operational parameters;*

344. Bonnell discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- g. *and performing a corrective action to fix any detected abnormalities;*

345. Bonnell discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

346. Bonnell discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

347. Bonnell discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

348. Bonnell discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

349. Bonnell discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

350. Bonnell discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

351. Bonnell discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

352. I expect to testify that Bonnell anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Bonnell discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-15.

**G. Anticipation by and/or Obviousness in View of United States Patent No. 6,122,664, issued to Boukobza, et al. on 9/19/2000.**

353. As explained in detail below and in the chart attached as Ex. B-15, Boukobza anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

354. Boukobza discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

Finally, relative to specific module “system”, it must be possible, among other things, to monitor and measure the cpu time, the disk space, the inputs/outputs, the storage, the exchange rate, the number of users, the pagination, the network, etc. Thus it is possible, for example, to measure the cpu utilization per second, with default display, or the input/output rate per second, or to identify the processors which are the largest users of cpu and collect them in order to perform an autonomous analysis (“offline”, tbc).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Boukobza at 34:43-51.

355. Further examples include Boukobza at 1:57-64, 2:21-38, 2:55-65, 4:36-42, 4:63-5:9, 5:63-6:14, Fig. 1.

b. *running at least one thread in a first runtime environment;*

356. Boukobza discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

*The processing of the conditions can involve several objects*, and in this case there is the question of knowing in which nodes the conditions are to be processed. In effect, a condition can be linked to a parameter of an object, and is therefore part of the description of this parameter. A parameter contains the description of its measurement (command to be executed, trace, curve display, etc.), the description of a certain number of single conditions related to the measurement just performed (operator, threshold, etc.) along with, for each condition, the action to be initiated when this condition is true. *A parameter is attached to an object which in turn is always associated with a node*, so the processing of the parameter and its conditions is carried out in the node associated with the object.

Boukobza at 5:19-32.

357. Further examples include Boukobza at 5:40-46, 7: 50-59.

358. To the extent this limitation is not expressly disclosed by Boukobza ’664, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

359. Boukobza discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

Remarkably, the autonomous agent installed in each node to be monitored is primarily composed of a generic agent *related to a plurality of specific modules, each of which is specific to an object type or to a particular domain*, of files containing the basic functions used, each node to be monitored also having its own files of parameters, conditions and associated actions for controlling its own monitoring and thus allowing as near as possible the processing of different types of objects by measuring the parameters, evaluating the conditions and initiating the actions linked to the conditions for all the objects described.

Boukobza at 4:5-15.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

360. Further examples include Boukobza at 5:19-32.

361. To the extent this limitation is not expressly disclosed by Boukobza, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

d. *detecting if there is an abnormality in the monitored operational parameters;*

362. Boukobza discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

In this way, in order to ensure effective monitoring of the applications running in the plurality of nodes, the process applied in this case makes it possible to measure specific parameters of each application, to *test conditions on these parameters* relative to thresholds, and *then to execute an action in order to warn of a problem*, to reconfigure or to correct.

Boukobza at 2:46-52.

363. Further examples include Boukobza at 8:50-67, 20:44-56, 34:52-60, 6:30-35, 21:43-56, 21:57-22:32.

e. *and performing a corrective action to fix any detected abnormalities,*

364. Boukobza discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

Each autonomous agent installed in each node (or machine), in addition to the parameter measurements it performs, the conditions it evaluates (in this node), the actions (reconfiguration, *correction*, alert) associated with these conditions it initiates or the operations it performs later, feeds back to the management node the information to be displayed, such as for example, the change of state of the objects, the parameter values to be displayed in curve form, etc.

Boukobza at 3:30-39.

365. Further examples include Boukobza at 2:46-52, 34:52-60, 21:43-56.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

366. Boukobza discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

The actions on condition can be explained as follows. An action is triggered if: a condition on one or more (status or metric) parameters is true, a multiple condition is true, a critical error appears in a "log" or "trace" file. For an action of a condition attached to a parameter, the parameters sent to the action are: "object\_id", "param\_id", "condition\_number", "param\_value", "threshold", "operator", "object\_specif\_string". For an action of a multiple condition, the parameters sent to the action are: "cond\_id". ***The action can be predefined or supplied by the administrator.*** It can include: a request to display the status ("ok", "down", "warning") of one or more "objects to be monitored" and the refreshing of the icon according to the status to be displayed. An "object to be monitored" is always displayed in the form of an icon. an immediate predefined correction, with or without a tuning request via the interface GUI. a predefined correction proposition, to be activated offline, a sending of a message to the administrator [interface GUI or history log of a product or knowledge base of an integrated systems management product, mail, audible signal ("beep"), etc.], ***a corrective action, completely described by the administrator.*** This can correspond to adding a Tuxedo server if the cpu utilization rate is no higher than x%, a correlation test on other data "ora/tux/syst", then if the latter is satisfied, action as above. Then requesting the value of another parameter ("last\_value", average of n values, "delta", "online" value), opening, on the management screen.

Boukobza at 14:27-63

367. Further examples include Boukobza at 8:50-67.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

368. Boukobza discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

The operation of the generic management agent occurs in the following way:

reading of its configuration file. Syntactic and semantic analysis ("MOD IF PARAM" of a non-existent parameter, etc.)

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

merging of the configuration file supplied by the administrator and the configuration file supplied by the specific modules.

sending of the resulting configuration file to each agent (*filtering relative to the objects to be monitored in the machine and taking into account the problems* inherent to the malfunctioning nodes or to the nodes provided for backing up the data of a malfunctioning node in another node).

Boukobza at 21:43-56.

369. Further examples include Boukobza at 21:57-22:32, 34:52-60.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

370. Boukobza discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

In this way, thanks to the idea of the invention, the *management node has excellent global visibility* since the present process, once the monitoring has been configured in a filtered way, (which means that a configuration is specific to an object and its environment and that therefore the same object can be configured differently depending on its situation and its distribution), makes it possible to monitor a plurality of object types, to give them unified functions, and also to correlate, among these functions, measurements on different types of objects.

Boukobza at 3:40-49.

371. Further examples include Boukobza at 2:21-38, 21:43-56.

372. To the extent this limitation is not expressly disclosed by Boukobza, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

2. **Claim 2**

- a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

373. Boukobza discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

The Tuxedo administrator assigns a weight to the services and uses it ***to equilibrate the load***. These weights are not necessarily well chosen, especially over time. These weights must be adjusted automatically as a function of administrative criteria.

Boukobza at 34:34-38.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

374. Boukobza discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

The present invention relates to a process for monitoring a plurality of object types of a plurality of nodes (N1, N2, ... , Nn) comprising a management node (MN) in an information system. Monitoring is configured and then distributed in a filtered way from the management node (MN) to autonomous agents (SAA), an autonomous agent being installed in each node to be monitored in order, by providing intertype correlation, either to locally process the different object types or all of the objects of a domain called a global object, defined generically, or to feed back information to be displayed to the graphical interface of the management node, each agent comprising a plurality of specific modules (SM1, SM2, ... , SMn) specific to the different object types or to a particular domain, each specific module measuring static and dynamic parameters particular to the object type it monitors and collecting said measurements, ***testing conditions on said parameters relative to predefined thresholds*** and possibly triggering actions associated with said tested conditions, which parameters, conditions and actions can be modified by the user of the management node.

Boukobza at Abstract.

375. Further examples include Boukobza at 2:39-52, 16:59-65, 31:16-20, Cl. 1.

376. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****4. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

377. Boukobza discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6.

378. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

**5. Claim 7**

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

379. Boukobza discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. For example:

Finally, relative to specific module "system", it must be possible, among other things, *to monitor and measure the cpu time, the disk space, the inputs/outputs, the storage, the exchange rate, the number of users, the pagination, the network*, etc. Thus it is possible, for example, to measure the cpu utilization per second, with default display, or the input/output rate per second, or to identify the processors which are the largest users of cpu and collect them in order to perform an autonomous analysis ("offline", tbc).

Boukobza at 34:43-51.

380. Further examples include Boukobza at 1:57-64, 2:21-38, 2:55-65, 4:36-42, 4:63-5:8, 5:63-6:14, Fig. 1.

- b. *running at least one thread in a first runtime environment;*

381. Boukobza discloses this element of claim 7, "running at least one thread in a first runtime environment," at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

382. Boukobza discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;and*

383. Boukobza discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *performing a corrective action to fix any detected abnormalities;*

384. Boukobza discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

385. Boukobza discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

386. Boukobza discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

387. Boukobza discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

388. Boukobza discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

389. Boukobza discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

The present invention relates to a process for monitoring a plurality of object types of a plurality of nodes (N1, N2, ... , Nn) comprising a management node (MN) in ***an information system***.

Boukobza at Abstract.

390. Further examples include Boukobza at 1:11-13, 1:57-64, Fig. 1.

- b. *a processor;*

391. Boukobza discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

The processing of the conditions can involve several objects, and in this case there is the question of knowing in which nodes the conditions are to be processed. In effect, a condition can be linked



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to a parameter of an object, and is therefore part of the description of this parameter. A parameter contains the description of its measurement (command to be executed, trace, curve display, etc.), the description of a certain number of single conditions related to the measurement just performed (operator, threshold, etc.) along with, for each condition, the action to be initiated when this condition is true. A parameter is attached to an object which in turn is always associated with a node, so *the processing of the parameter and its conditions is carried out in the node associated with the object*. Boukobza at 5:19-32.

392. Further examples include Boukobza at 6:55-61.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

393. Boukobza discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

The present invention relates to a process for monitoring a plurality of object types of a plurality of nodes (N1, N2, ... , Nn) comprising a management node (MN) in *an information system*.

Boukobza at Abstract.

394. Further examples include Boukobza at 1:11-13, 1:57-64, Fig. 1.

- d. *running at least one thread in a first runtime environment;*

395. Boukobza discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

396. Boukobza discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

- f. *detecting there is an abnormality in the monitored operational*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

397. Boukobza discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

398. Boukobza discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

399. Boukobza discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

400. Boukobza discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

401. Boukobza discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

402. Boukobza discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

403. Boukobza discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

404. Boukobza discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

405. I expect to testify that Boukobza anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Boukobza discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-16.

**H. Anticipation by and/or Obviousness in View of United States Patent No. 6,665,262, filed 2/16/1999 and issued to Lindskog, et al. on 12/16/2003.**

406. As explained in detail below and in the chart attached as Ex. B-18, Lindskog anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. ***Claim 1***

a. *A computer-implemented method, comprising:*

407. Lindskog discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example,

The present invention comprises a system and ***method for performing distributed fault management functions in a communications network.***

Lindskog at 3:11-14.

408. Further examples include Lindskog at Abstract, 4:43-57, Fig. 1.

b. *running at least one thread in a first runtime environment;*

409. Lindskog discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example,

A system and method for distributing the fault management functions throughout a communications network. ***Each node in the network includes an associated fault agent/ configuration agent pair.*** The fault agent receives alarm information and correlates the alarm information to identify a cause of a fault in the network.

Lindskog at Abstract.

410. Further examples include Lindskog at 4:58-5:1, Fig. 1.

411. To the extent this limitation is not expressly disclosed by Lindskog, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

412. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

413. Lindskog discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example,

Referring now to FIG. 1, there is illustrated a simplified diagram of the distributed fault management system 10 of the present invention. ***The system 10 includes any number of fault agents (FA) 12 and configuration agents (CA) 14 operating on different levels. Usually, the FAs 12 and CAs***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

***14 are arranged in pairs.*** These FA-CA pairs 16 can be associated with a particular device, such as a radio network controller or radio network manager in a cellular system, or with a logical entity, such as a cell-pair or a location area. ***Each FA-CA pair 16 supervises one or more underlying network resources 18 and/or subordinate FA-CA pairs 16.***

Lindskog at 4:58-5:1.

414. Further examples include Lindskog at 5:6-17, 10:63-11:4, 10:63-11:4, 8:41-58, 10:19-34, 7:63-8:16, Fig.1, Fig. 2, Fig. 3.

415. To the extent this limitation is not expressly disclosed by Lindskog, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

416. To the extent this limitation is not expressly disclosed by Lindskog, it would have been obvious to a person of ordinary skill in the art in view of Cantrill and/or U.S. Patent No. 6,658,654 (“Berry ’654”). See Berry ’654 at 1:18-36, 3:42-64, 12:65-13:10, Fig. 6A. See also Cantrill at p. 253, sec. 1, Cantrill at p. 260, sec. 5.2, Cantrill at Fig. 2.

d. *detecting if there is an abnormality in the monitored operational parameters;*

417. Lindskog discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example,

Referring now to FIG. 1, there is illustrated a simplified diagram of the ***distributed fault management system*** 10 of the present invention. The system 10 includes any number of fault agents (FA) 12 and configuration agents (CA) 14 operating on different levels. Usually, the FAs 12 and CAs 14 are arranged in pairs. These FA-CA pairs 16 can be associated with a particular device, such as a radio network controller or radio network manager in a cellular system, or with a logical entity, such as a cell-pair or a location area. Each FA-CA pair 16 supervises one or more underlying network resources 18 and/or subordinate FA-CA pairs 16.

Lindskog at 4:58-5:1.

418. Further examples include Lindskog at 5:6-14, 3:36-50, 5:42-55, 6:40-65, 10:24-34, 3:64-4:10, 9:22-31, 10:63-11:4, 11:47-56, Fig. 1.

e. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities,*

419. Lindskog discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example,

*Once the alarm data reaches a fault agent at which the underlying fault can be handled, the fault agent forwards the fault information to the associated configuration agent. The configuration agent processes the fault information to generate reconfiguration data for correcting or otherwise reducing the effect of the fault and sends the reconfiguration data to a network resource (ordering it to perform some action), or to at least one subordinate configuration agent, depending on which nodes are affected by the proposed reconfiguration. Each subordinate configuration agent that receives the reconfiguration data generally performs further processing to generate more detailed reconfiguration data, which is then passed on to even lower level configuration agents or to underlying network resources. This process repeats until the reconfiguration is fully implemented.*

Lindskog at 3:36-50.

420. Further examples include Lindskog at 10:24-34, 5:42-55, 3:64- 4:10, 8:41-58, 11:47-56, Fig. 2.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

421. Lindskog discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example,

*In particular, when a fault agent receives alarm data, either from a subordinate fault agent or from a network resource, the fault agent analyzes the received alarm data to identify a cause of the alarm and to determine if the underlying fault that caused the alarm can be handled at the current node. If not, then the fault agent produces a new alarm, which summarizes the received alarm data, and passes the new alarm to an interconnected fault agent.*

Lindskog at 3:27-35.

422. Further examples include Lindskog at 5:6-14, 3:36-50, Fig. 4.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

423. Lindskog discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example,

In this manner, the CAs 14 in the system 10 are able to receive a high level cell plan, generate a new, more detailed cell plan for the subordinate CAs 14 and underlying resources 18, and transmit the new cell plan to the appropriate subordinate CAs 14 and resources 18. ***Similarly, with respect to received events that contain alarm data, the CAs 14 are able to evaluate the alarm, generate appropriate cell plans or other corrective measures, and transmit appropriate instructions to the subordinate CAs 14 and underlying resources 18 (as indicated at 48).***

Lindskog at 10:24-34.

424. Further examples include Lindskog at 3:64-4:10, 3:36-50, 5:42-65.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

425. Lindskog discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example,

In particular, ***when a fault agent receives alarm data, either from a subordinate fault agent or from a network resource,*** the fault agent analyzes the received alarm data to identify a cause of the alarm and to determine if the underlying fault that caused the alarm can be handled at the current node. If not, then the fault agent produces a new alarm, which summarizes the received alarm data, and passes the new alarm to an interconnected fault agent.

Lindskog at 3:27-35.

426. Further examples include Lindskog at 4:58-5:1, 5:6-14, 5:18-41, 3:36-50, 8:41-58, Fig. 2, Fig 4.

427. To the extent this limitation is not expressly disclosed by Lindskog ’262, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

428. To the extent this limitation is expressly disclosed by Linskog '262, it would have been obvious in view of Turek '070. *See* Ex. B-1 at limitation 1.7.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

429. Linskog discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example,

In particular, the distributed fault management architecture the present invention ***decentralizes the fault management operations*** and increases the robustness of the fault management system, while permitting faults to be addressed in real time. ***Essentially, fault management capabilities are distributed throughout the various levels of a communications network. Maximum efficiency is then achieved by performing fault processing, and making necessary corrections, at the lowest possible level. In addition, the decentralization of the fault management operations allows scalability (i.e., the ability to continue operations as new equipment or levels are added to the network)*** of the system to a much larger degree than with a centralized fault management solution.

Linskog at 4:43-57.

430. Further examples include Linskog at 4:58-5:1, 11:56-62, Fig. 1.

431. To the extent this limitation is not expressly disclosed by Linskog '262, it would have been obvious to a person of ordinary skill in the art to perform corrective actions that included load balancing operations in a distributed networking system at least to improve network efficiency and scalability.

432. To the extent this limitation is not expressly disclosed by Linskog '262, it would have been obvious to a person of ordinary skill in the art in view of Buchanan. *See* Buchanan at Abstract, Buchanan at sec. 6, Buchanan at sec. 1, Buchanan at sec. 4, Buchanan at Fig. 1. *See also* Cantrill at p. 253, sec. 1, Cantrill at p. 260, sec. 5.2, Cantrill at Fig. 2.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

433. Linskog discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example,

Returning now to FIG. 2, ***when an update condition 108 or 114 is triggered in the event generator 44, the event generator 44 sends a message (as indicated at 46) to the event database 38, causing the event record for the generated event to be updated by changing the value of the value field and storing a new time in the time generated field.*** Whenever an event of an FA 12 is updated in the event database 38, the event dispatcher 40 is activated and the events are queued, as discussed above in connection with the subordinate FAs 12, one after another according to their priority and deadline. Thus, when Event n is generated in the FA 11, it is transmitted via the event dispatcher to the requesting agent CA 11, in accordance with the original subscription request (which was sent as indicated at 30). In this manner, the appropriate CAs 14 in the system 10 are able to request and receive alarm information (events), in response to which the receiving CA14 is potentially able to take the necessary corrective actions.

Linskog at 8:41-58.

434. Further examples include Linskog at 6:40-65, 5:18-41, Fig. 2.

435. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

436. Linskog discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example,

Each subordinate configuration agent that receives the reconfiguration data generally performs further processing to generate more detailed reconfiguration data, which is then passed on to even lower level configuration agents or to underlying network resources. This process repeats until the reconfiguration is fully implemented.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Lindskog at 3:44-50.

437. Further examples include Lindskog at Abstract, 5:42-55.

438. To the extent this limitation is not expressly disclosed by Lindskog '262, it would have been obvious to a person of ordinary skill in the art to utilize well known Dijkstra's Self Stabilization Algorithm in a distributed networking system at least to improve system efficiency.

439. To the extent this limitation is not expressly disclosed by Lindskog '262, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

440. Further, it would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in a distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek.

441. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

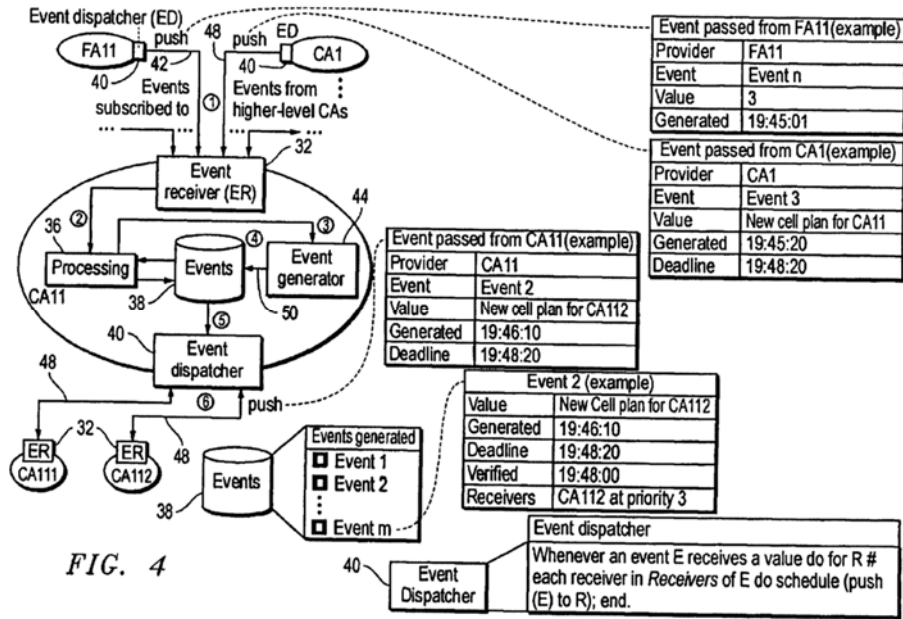
5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

442. Lindskog discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example,



b. *running at least one thread in a first runtime environment;*

443. Lindskog discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

444. Lindskog discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

445. Lindskog discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

446. Linskog discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

447. Linskog discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

448. Linskog discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**6. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

449. Linskog discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

**7. Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

450. Linskog discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

a. *A system, comprising:*

451. Linskog discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example,

The present invention comprises a ***system*** and method ***for performing distributed fault management functions in a communications network.***

Linskog at 3:11-14.

452. Further examples include Linskog at Abstract, 4:43-57, 4:58-5:1, 5:6-14, 11:56-67, Fig. 1.

b. *a processor;*

453. Linskog discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example,

When an event is received by CA11, the event is immediately passed on to a processing unit 36 in CA11. ***The processing unit*** 36 checks to ensure that the event is valid (e.g., by accessing the events database 38), and if so, the event is forwarded to the event generator 44. The event generator 44 is responsible for carrying out all computations needed to submit proper configuration changes to the underlying CAs 14 and network resources 18. These computations can be relatively complex, especially for the generation or extraction of a new cell plan.

Linskog at 9:22-31.

454. Further examples include Linskog at 6:66-7:6, Claim 5, 12:48-51, Fig. 2.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

455. Linskog discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Similarly, the configuration agents include an event receiver for receiving both fault information from associated fault agents and high level reconfiguration data from higher level configuration agents. The configuration agents further include an event generator for generating the reconfiguration data necessary to reduce the effect of a detected fault and for generating more detailed reconfiguration data from the high level data that is received from supervising configuration agents. Once the reconfiguration data is generated, the event generator updates the ***configuration information that is stored in an event database***, and, as a result, the event dispatcher sends the updated configuration information to the subordinate configuration agents and/or to underlying network resources for implementation.

Lindskog at 3:64-4:10.

456. Further examples include Lindskog at 9:14-21, 9:51-67, Fig. 4.

457. To the extent this limitation is not expressly disclosed by Lindskog, it would have been obvious to a person of ordinary skill in the art, because it was well known to use a memory coupled to the processor to store instructions to be executed by the processor.

d. *running at least one thread in a first runtime environment;*

458. Lindskog discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

459. Lindskog discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

460. Lindskog discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

461. Lindskog discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

462. Lindskog discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

463. Lindskog discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

464. Lindskog discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*load balancing operation.*

465. Lindskog discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

466. Lindskog discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

467. Lindskog discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

468. I expect to testify that Lindskog anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Lindskog discloses each and every limitation of those claims is included as Ex. B-18.

**I. Anticipation by and/or Obviousness in View of United States Patent No. 6,675,128, filed 9/30/1999 and issued to Hellerstein, et al. on 1/6/2004.**

469. As explained in detail below and in the chart attached as Ex. B-20, Hellerstein anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

470. Hellerstein discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Referring now to FIG. 3, a block diagram is shown of an illustrative hardware implementation for both a managed element (e.g., 100-1 through 100-N) and a manager 140 according to the invention. ***In one embodiment, a managed element 100 and a manager 140 may be respectively implemented in individual computer systems.*** Each computer system may include a processor 300 coupled to a memory 310 and I/O device(s) 320, as shown in FIG. 3.

Hellerstein at 7:51-59.

471. Further examples include Hellerstein at Figs. 1, 3.

b. *running at least one thread in a first runtime environment;*

472. Hellerstein discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

Referring now to FIG. 3, a block diagram is shown of an illustrative hardware implementation for both a managed element (e.g., 100-1 through 100-N) and a manager 140 according to the invention. ***In one embodiment, a managed element 100 and a manager 140 may be respectively implemented in individual computer systems.*** Each computer system may include a processor 300 coupled to a memory 310 and I/O device(s) 320, as shown in FIG. 3.

Hellerstein at 7:51-59.

473. Further examples include Hellerstein at Figs. 1, 3.

474. To the extent this limitation is not expressly disclosed by Hellerstein, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

475. Hellerstein discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

Generally, existing practice for detecting performance and availability problems consists of the following steps: (i) determining a set of metrics to monitor that indicate the presence of problems (e.g., CPU utilization, error rates, transaction request rates); (ii) establishing thresholds on the values of these metrics based on past experience; (iii) using management system software to detect threshold violations; and (iv) responding to threshold violations by taking actions (e.g., adjusting user priorities, restricting the admission of traffic into the network).



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Hellerstein at 1:20-29.

476. Further examples include Hellerstein at 6:28-36.

477. To the extent this limitation is not expressly disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

d. *detecting if there is an abnormality in the monitored operational parameters;*

478. Hellerstein discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

Typically, policies are expressed as if-then rules. The if-part (or left-hand side, LHS, of the policy) contains a predicate expressed as a bound on one or more metrics. The then-part (or right-hand side, RHS, of the policy) contains the action to take. ***An example is: "If CPU utilization is greater than 90%, then alarm."*** Here, "CPU utilization" is the metric, the relational operator is “greater than,” the threshold value is “90%,” and the action is "alarm" (e.g., send an urgent message to the operations console). The threshold value for alarms may be chosen so that it lies well beyond what is considered normal. We use the term "alarm threshold" for the metric value that, if exceeded, results in either the generation of an alarm or a management action (e.g., terminate a process). Existing approaches check for threshold violations and, when these violations occur, initiate the action specified in the right-hand side of the policy.

Hellerstein at 1:37-52.

479. Further examples include Hellerstein at 4:52-5:18.

e. *and performing a corrective action to fix any detected abnormalities,*

480. Hellerstein discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

Typically, policies are expressed as if-then rules. The if-part (or left-hand side, LHS, of the policy) contains a predicate expressed as a bound on one or more metrics. The then-part (or right-hand side, RHS, of the policy) contains the action to take. ***An example is: "If CPU utilization is greater than 90%, then alarm."*** Here, “CPU utilization” is the metric, the relational operator is “greater than,” the threshold value is “90%,” ***and the action is "alarm" (e.g., send an urgent message to the operations console).*** The threshold value for alarms may be chosen so that it lies well beyond what is considered normal. We use the term “alarm threshold” for the metric value that, if exceeded, results in either the generation of an alarm or a

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

management action (e.g., terminate a process). Existing approaches check for threshold violations and, when these violations occur, initiate the action specified in the right-hand side of the policy.

Hellerstein at 1:37-52.

481. Further examples include Hellerstein at 4:52-5:18.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

482. Hellerstein discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Referring now to FIG. 2, a block diagram illustrating a model-based policy agent 120 according to an exemplary embodiment of the present invention is shown. In the model-based policy agent, ***an agent side policy controller 200 provides overall control and interacts with the platform services to update local copies of policy repositories*** and to report alarms and warnings to the manager 140.

Hellerstein at 7:10-16.

483. Further examples include Hellerstein at Fig. 2.

484. To the extent this limitation is not expressly disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time that an agent or its runtime environment could be programmed to request a corrective policy from an entity outside of the particular runtime environment.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

485. Hellerstein discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Typically, policies are expressed as if-then rules. The if-part (or left-hand side, LHS, of the policy) contains a predicate expressed as a bound on one or more metrics. The then-part (or right-hand side, RHS, of the policy)

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

contains the action to take. *An example is: "If CPU utilization is greater than 90%, then alarm."* Here, "CPU utilization" is the metric, the relational operator is "greater than," the threshold value is "90%," *and the action is "alarm" (e.g., send an urgent message to the operations console).* The threshold value for alarms may be chosen so that it lies well beyond what is considered normal. We use the term "alarm threshold" for the metric value that, if exceeded, results in either the generation of an alarm or a management action (e.g., terminate a process). Existing approaches check for threshold violations and, when these violations occur, initiate the action specified in the right-hand side of the policy.

Hellerstein at 1:37-52.

486. Further examples include Hellerstein at 4:52-5:18.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

487. Hellerstein discloses this element of claim 1, "wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads." '659 Pat. at Cl. 1. For example:

On each managed element is also a model-based policy agent 120 that enforces performance management policies, as will be explained in detail below. The managed elements communicate with *a manager 140 that is controlled by an administrative user 160 who is responsible for authoring and ensuring the distribution of management policies through the policy authoring, distribution, and reporting component 145.* In particular, the manager 140 provides an administrator 160 a place to enter information into the following repositories: model reconstruction policies 205, alarm policies 225, control policies 210, and action trigger policies 220.

Hellerstein at 6:37-48.

488. Further examples include Hellerstein at Fig. 1.

489. To the extent this limitation is not expressly disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**2. *Claim 2*

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

490. Hellerstein discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

Generally, existing practice for detecting performance and availability problems consists of the following steps: (i) determining a set of metrics to monitor that indicate the presence of problems (e.g., CPU utilization, error rates, transaction request rates); (ii) establishing thresholds on the values of these metrics based on past experience; (iii) using management system software to detect threshold violations; and (iv) responding to threshold violations by taking actions (e.g., adjusting user priorities, restricting the admission of traffic into the network).

Hellerstein at 1:20-29.

491. To the extent this limitation is not expressly disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art to perform corrective actions that included load balancing operations in a distributed networking system at least to improve network efficiency and scalability.

3. *Claim 3*

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

492. Hellerstein discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

Typically, policies are expressed as if-then rules. The if-part (or left-hand side, LHS, of the policy) contains a predicate expressed as a bound on one or more metrics. The then-part (or right-hand side, RHS, of the policy) contains the action to take. **An example is: "If CPU utilization is greater than 90%, then alarm."** Here, "CPU utilization" is the metric, the relational operator is "greater than," the threshold value is "90%," and the action is "alarm" (e.g., send an urgent message to the operations console). The threshold value for alarms may be chosen so that it lies well beyond what is considered normal. We use the term "alarm threshold" for the metric value that, if exceeded, results in either the generation of an alarm or a management action (e.g., terminate a process). **Existing approaches check**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*for threshold violations and, when these violations occur, initiate the action specified in the right-hand side of the policy.*

Hellerstein at 1:37-52.

493. Further examples include Hellerstein at 4:52-5:18, 7:10-21, Fig. 2.

494. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

495. Hellerstein discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example:

Typically, policies are expressed as if-then rules. The if-part (or left-hand side, LHS, of the policy) contains a predicate expressed as a bound on one or more metrics. The then-part (or right-hand side, RHS, of the policy) contains the action to take. ***An example is: "If CPU utilization is greater than 90%, then alarm."*** Here, "CPU utilization" is the metric, the relational operator is "greater than," the threshold value is "90%," ***and the action is "alarm" (e.g., send an urgent message to the operations console).*** The threshold value for alarms may be chosen so that it lies well beyond what is considered normal. We use the term "alarm threshold" for the metric value that, if exceeded, results in either the generation of an alarm or a management action (e.g., terminate a process). Existing approaches check for threshold violations and, when these violations occur, initiate the action specified in the right-hand side of the policy.

Hellerstein at 1:37-52.

496. Further examples include Hellerstein at 4:52-5:18.

497. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

498. Hellerstein discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

Referring now to FIG. 3, a block diagram is shown of an illustrative hardware implementation for both a managed element (e.g., 100-1 through 100-N) and a manager 140 according to the invention. ***In one embodiment, a managed element 100 and a manager 140 may be respectively implemented in individual computer systems.*** Each computer system may include a processor 300 coupled to a memory 310 and I/O device(s) 320, as shown in FIG. 3.

Hellerstein at 7:51-59.

499. Further examples include Hellerstein at Figs. 1, 3.

b. *running at least one thread in a first runtime environment;*

500. Hellerstein discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

501. Hellerstein discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

502. Hellerstein discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

503. Hellerstein discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

504. Hellerstein discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

505. Hellerstein discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**6. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

506. Hellerstein discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

**7. Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

507. Hellerstein discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. ***Claim 13***

a. *A system, comprising:*

508. Hellerstein discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

Referring now to FIG. 3, a block diagram is shown of an illustrative hardware implementation for both a managed element (e.g., 100-1 through 100-N) and a manager 140 according to the invention. ***In one embodiment, a managed element 100 and a manager 140 may be respectively implemented in individual computer systems.*** Each computer system may include a processor 300 coupled to a memory 310 and I/O device(s) 320, as shown in FIG. 3.

Hellerstein at 7:51-59.

509. Further examples include Hellerstein at Figs. 1, 3.

b. *a processor;*

510. Hellerstein discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

Referring now to FIG. 3, a block diagram is shown of an illustrative hardware implementation for both a managed element (e.g., 100-1 through 100-N) and a manager 140 according to the invention. ***In one embodiment, a managed element 100 and a manager 140 may be respectively implemented in individual computer systems.*** Each computer system may include a processor 300 coupled to a memory 310 and I/O device(s) 320, as shown in FIG. 3.

Hellerstein at 7:51-59.

511. Further examples include Hellerstein at Figs. 1, 3.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

512. Hellerstein discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

Referring now to FIG. 3, a block diagram is shown of an illustrative hardware implementation for both a managed element (e.g., 100-1 through 100-N) and a manager 140 according to the invention. ***In one embodiment, a managed element 100 and a manager 140 may be respectively implemented in individual computer systems.*** Each computer system may



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

include a processor 300 coupled to a memory 310 and I/O device(s) 320, as shown in FIG. 3.

Hellerstein at 7:51-59.

513. Further examples include Hellerstein at Figs. 1, 3.

d. *running at least one thread in a first runtime environment;*

514. Hellerstein discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

515. Hellerstein discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

516. Hellerstein discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

517. Hellerstein discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

518. Hellerstein discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

519. Hellerstein discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

520. Hellerstein discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**9. Claim 14**

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

521. Hellerstein discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**10. Claim 15**

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

522. Hellerstein discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

523. Hellerstein discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

524. I expect to testify that Hellerstein anticipates and/or renders obvious each Asserted Claim of the '659 Patent. A claim chart illustrating that Hellerstein discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-20.

**J. Anticipation by and/or Obviousness in View of *Distributed Management with Mobile Components*, by Kasteleijn, et al., published in the May 1999 Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated Network Management.**

525. As explained in detail below and in the chart attached as Ex. B-2, Kasteleijn anticipates and renders obvious the claims of the '659 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

526. Kasteleijn discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” '659 Pat. at Cl. 1. For example:

The distributed management framework (DMF) presented in this paper provides an environment which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates. We present an object-oriented, Java-based implementation of the DMF and describe applications developed on this platform.

Kasteleijn at 857.

- b. *running at least one thread in a first runtime environment;*

527. Kasteleijn discloses this element of claim 1, “running at least one thread in a first runtime environment.” '659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The distributed management architecture presented in this paper provides an *environment* which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, *management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates.* Management tasks are supported by a distributed directory and secure communications. The architecture does not force the programmer to design its management applications according to any given programming paradigm. Rather, the programmer is free to use paradigms such as client-server and cooperating peers. Mixed solutions are also possible.

Kastaleijin at 858.

528. Further examples include Kasteleijn at 858, 859–860, 861, 866, 864.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

529. Kasteleijn discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

In our scenario shown in Figure 4, the EnterpriseManager listens for "delay threshold exceeded at LAN NYC" events from other MCs in the DMF. When it receives such an event, the following set of actions will take place:

- The EnterpriseManager creates/activates a *TrafficAnalyzer MC which monitors traffic on a LAN segment and measures IP traffic* in terms of a selected set of application protocols. The TrafficAnalyzer receives from the Enterprise Manager a TaskDescription object (1). *This TaskDescription "defines" the task, which means that it specifies the subnet to monitor, types of application protocols that need to be observed (e.g., HTTP, SMTP, FTP) and the monitoring duration.*

Kasteleijn at 866.

530. Further examples include Kasteleijn at 866, 867-868.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

531. Kasteleijn discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

#### **4. Implementation Example: Enterprise Manager**

In this section, we present a management application called EnterpriseManager that runs on our DMF implementation. EnterpriseManager is a management component that runs on a DMN

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

(DMN-ZRH), and is *used to analyze IP traffic characteristics when assigned performance thresholds are exceeded*.

Kasteleijn at 866.

532. Further examples include Kasteleijn at 866–867.

- e. *and performing a corrective action to fix any detected abnormalities,*

533. Kasteleijn discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

Management tasks can be assigned some level of autonomy, so that functions such as discovery can *scale and adapt to the managed network* without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 860.

534. Further examples include Kasteleijn at 861.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

535. Kasteleijn discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example, Kasteleijn discloses that an MC may request further code, MCs, or management functions provided by MCs and their associated policies where required (*i.e.*, where it cannot perform its task without these components).

536. As another example:

An MC may act as a management console, *integrating management functions provided by other MCs active within the DMF*. A more sophisticated MC may carry out a longer running task such as collecting traffic statistics. Another type of MC can be an *adaptive* agent which travels to a DMN with an assigned management task, discovers the management tools available at that DMN relevant to its task, and then adapts and/or composes a solution strategy for the given environment in order to efficiently carry out its task.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Kasteleijn at 861.

537. Further examples include Kasteleijn at 862, 863, 865, 869.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

538. Kasteleijn discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Management tasks can be assigned some level of autonomy, so that functions such as discovery can *scale and adapt to the managed network* without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 860.

539. Further examples include Kasteleijn at 861.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

540. Kasteleijn discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

## **2. Distributed Management Framework**

Distributed Management Framework (DMF) provides an object-oriented architecture where lightweight, mobile management applications can be dynamically deployed for distributed management. As shown in Figure 2, the Distributed Management Framework consists of a collection of *distributed management nodes (DMNs), where each DMN provides the execution environment and supporting services for management components (MCs), i.e., the management applications.* The DMNs form a network of peers over which hierarchies or cooperating sets of management components can be run. A *distributed directory*, maintained by each DMN is used in locating management components and services. Components and configuration of the DMF are stored in a set of replicated *code and configuration stores (CCSs)*, from where they can be downloaded to the DMNs as required. In the following, we describe these components in detail.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Kasteleijn at 860.

541. Further examples include Kasteleijn at 859–860, 861–862, 866, Figs. 1-3.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

542. Kasteleijn discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. To the extent this limitation is not met by Kasteleijn, a person of ordinary skill in the art would have found this limitation obvious.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

543. Kasteleijn discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

In our scenario shown in Figure 4, the EnterpriseManager listens for "***delay threshold exceeded at LAN NYC" events from other MCs in the DMF.***" When it receives such an event, the following set of actions will take place:

- The EnterpriseManager creates/activates a TrafficAnalyzer MC which monitors traffic on a LAN segment and measures IP traffic in terms of a selected set of application protocols. The TrafficAnalyzer receives from the Enterprise Manager a TaskDescription object (1). This TaskDescription "defines" the task, which means that it specifies the subnet to monitor, types of application protocols that need to be observed (e.g., HTTP, SMTP, FTP) and the monitoring duration.

Kasteleijn at 866.

544. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Algorithm.*

545. Kasteleijn discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6.

546. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

547. Kasteleijn discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

The distributed management framework (DMF) presented in this paper provides an environment which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates. We present an object-oriented, Java-based implementation of the DMF and describe applications developed on this platform.

Kasteleijn at 857.

- b. *running at least one thread in a first runtime environment;*

548. Kasteleijn discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

549. Kasteleijn discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters;*

550. Kasteleijn discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

551. Kasteleijn discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

552. Kasteleijn discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

553. Kasteleijn discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**6. Claim 8**

a. *The computer-readable medium of claim 7, wherein the corrective*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

554. Kasteleijn discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. **Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

555. Kasteleijn discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. **Claim 13**

- a. *A system, comprising:*

556. Kasteleijn discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

The distributed management framework (DMF) presented in this paper provides an environment which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates. We present an object-oriented, Java-based implementation of the DMF and describe applications developed on this platform.

Kasteleijn at 857.

- b. *a processor;*

557. Kasteleijn discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

The management application begins by *creating and then sending a discovery agent D to the IP router as shown in Figure 1 (1, 2)*. From the router configuration data, agent D discovers the different subnets adjacent to the router. *It then creates an NFS query agent N per subnet (3). Each such agent visits all hosts on the assigned subnet looking for NFS daemons and if one is present, collects status information (4)*. Once the collection is complete, each NFS query agent returns to the originating discovery agent D to deliver its consolidated data from the subnet and

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

terminates (5). Agent D waits until all NFS agents have reported back after which it returns to the management application with the results (6, 7).

The simple example described above demonstrates a number of goals that we want to achieve with our architecture:

- The execution of the three major tasks, topology discovery, NFS daemon discovery and NFS status determination are distributed, carried out by lightweight management tasks. There is no single processing bottleneck.
- Management tasks are executed close to the managed resources. Local consolidation of raw data results in reduced network management traffic on the network.
- A management task is *brought* to the managed resource, ***adapted to run in the environment of the resource and then executed***, rather than requiring a large set of pre-installed tools. Once the management task completes its assigned objectives, it leaves the managed resource to continue its processing elsewhere or terminates.
- Management tasks can be assigned some level of autonomy, so that functions such as discovery can scale and adapt to the managed network without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 859–860.

558. Further examples include Kasteleijn at Fig. 1.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

559. Kasteleijn discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

The management application begins by ***creating and then sending a discovery agent D to the IP router as shown in Figure 1 (1, 2)***. From the router configuration data, agent D discovers the different subnets adjacent to the router. ***It then creates an NFS query agent N per subnet (3). Each such agent visits all hosts on the assigned subnet looking for NFS daemons and if one is present, collects status information (4)***. Once the collection is complete, each NFS query agent returns to the originating discovery agent D to deliver its consolidated data from the subnet and terminates (5). Agent D waits until all NFS agents have reported back after which it returns to the management application with the results (6, 7).

The simple example described above demonstrates a number of goals that we want to achieve with our architecture:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- The execution of the three major tasks, topology discovery, NFS daemon discovery and NFS status determination are distributed, carried out by lightweight management tasks. There is no single processing bottleneck.
- Management tasks are executed close to the managed resources. Local consolidation of raw data results in reduced network management traffic on the network.
- A management task is *brought* to the managed resource, ***adapted to run in the environment of the resource and then executed***, rather than requiring a large set of pre-installed tools. Once the management task completes its assigned objectives, it leaves the managed resource to continue its processing elsewhere or terminates.
- Management tasks can be assigned some level of autonomy, so that functions such as discovery can scale and adapt to the managed network without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 859–860.

560. Further examples include Kasteleijn at Fig. 1.

d. *running at least one thread in a first runtime environment;*

561. Kasteleijn discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

562. Kasteleijn discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

563. Kasteleijn discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

564. Kasteleijn discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

565. Kasteleijn discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

566. Kasteleijn discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

567. Kasteleijn discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*load balancing operation.*

568. Kasteleijn discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

569. Kasteleijn discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

570. Kasteleijn discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

571. I expect to testify that Kasteleijn anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Kasteleijn discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-2.

**K. Anticipation by and/or Obviousness in View of *Proactive Management of Computer Networks using Artificial Intelligence Agents and Techniques*, by da Rocha, et al., published in the 1997 issue of Integrated Network Management V (Springer).**

572. As explained in detail below and in the chart attached as Ex. B-6, Da Rocha anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. ***Claim 1***

a. *A computer-implemented method, comprising:*

573. Da Rocha discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

574. Further examples are also found at da Rocha at 613, 619; *see generally, e.g.,* da Rocha at Sections 1, 2.

b. *running at least one thread in a first runtime environment;*

575. Da Rocha discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

To solve the problems associated with network management, the ISO, by OSI/NM, proposes three models: the Organizational Model, which establishes a hierarchy among management systems within one management environment, dividing the environment to be managed into various domains; the Informational Model, which defines the objects of management, their interrelationships and the operations made upon these objects. A MIB is needed to store managed objects; the Functional Model, which describes the functions - error, configuration, performance, account and security - of management. In this way, the concept of network management provides the administrator with sufficient means by which to distribute the network's resources to its users, while, due to the quantity of information available, allowing for proactive management.

da Rocha at 611.

576. Further examples include da Rocha at 613. *See generally, e.g.,* da Rocha at Sections 3, 4.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

577. To the extent this limitation is not expressly disclosed by da Rocha, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

578. Da Rocha discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

The task of monitoring and observing the network is best achieved by agents with resident action in the machines to be managed. In view of this, ***Agent 6 was created with the intent of measuring quality and operation in the network communication services by monitoring the volume of traffic and verifying the number of errors. These characteristics will later be used as a database for establishing a baseline.***

da Rocha at 613.

579. Further examples include da Rocha at 616, 618; da Rocha at Figs. 4, 6. *See generally, e.g.,* da Rocha at Sections 3, 4.

580. To the extent this limitation is not expressly disclosed by da Rocha, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

581. Da Rocha discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

This work was developed in the area of Computer Network Management. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. ***The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks,*** and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

da Rocha at Abstract.

582. Further examples include da Rocha at 618, 620, 621; da Rocha at Figs. 4, 6. *See generally, e.g.,* da Rocha at Sections 3, 4, 5.

e. *and performing a corrective action to fix any detected abnormalities,*

583. Da Rocha discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

Having these data, the following workflow was determined: run agent 6, Hostmem, Hostif and Hostperf in some sub-network nodes, measuring congestion and other statistics in different hours; based on the results, establish a baseline or base with known points as a measure of standard deviation for measurements in normal situations; implant the diagnostic model, so as to activate rules each time the measurement taken is beyond the standard parameters contained in the baseline, stipulated as: for each measurement taken from the monitoring agents that arrives within reach of the timed measurement in which it occurred, a diagnostic module will be triggered to verify whether or not problems exist according to the module's rules; if the diagnostic module verifies the a problem exists which could result in a performance drop or congestion, rules will be used to determine the motives for the event; in a third moment, after verifying the previous, measures are taken to avoid the problem, ***reporting the anomalies found to the network administrator and suggesting corrective measures.***

da Rocha at 613.

584. Further examples include da Rocha at 613; da Rocha at Fig. 6. *See generally, e.g.,* da Rocha at Abstract; Sections 2, 3.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

585. Da Rocha discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Having these data, the following workflow was determined: run agent 6, Hostmem, Hostif and Hostperf in some sub-network nodes, measuring congestion and other statistics in different hours; based on the results, establish a baseline or base with known points as a measure of standard deviation for measurements in normal situations; implant the diagnostic

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

model, so as to activate rules each time the measurement taken is beyond the standard parameters contained in the baseline, stipulated as: for each measurement taken from the monitoring agents that arrives within reach of the timed measurement in which it occurred, a diagnostic module will be triggered to verify whether or not problems exist according to the module's rules; if the diagnostic module verifies the a problem exists which could result in a performance drop or congestion, rules will be used to determine the motives for the event; in a third moment, after verifying the previous, measures are taken to avoid the problem, ***reporting the anomalies found to the network administrator and suggesting corrective measures.***

da Rocha at 613.

586. Further examples include da Rocha at 613, 618; da Rocha at Fig. 6. *See generally*, e.g., da Rocha at Abstract; Sections 2, 3.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

587. Da Rocha discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example, users can agree or disagree with the agent’s suggested actions (a corrective policy). This updates the agent’s knowledge base, and changes how the agent’s inference machine makes decisions. Thus, when the agent’s inference machine runs through its logic after the user has updated the knowledge base, the agent is applying the corrective policy:

***The knowledge base stores the knowledge of the expert and differentiates from a conventional database in that it is active in nature, permitting updates conforming to the context.*** The structure of the knowledge base will depend on the type of knowledge represented. To have deductive knowledge, the base will usually be composed of rules. To have modeling of physical structures, causal links or interrelationship between models, the ideal structure may be a semantic network. . . .

In a [sic] expert system, the knowledge of the problem's domain is organized separately from the other system knowledge, such as the procedures or steps for problem solving, or interaction with the user represented by the explanatory interface, which defines how to present the knowledge. This division is intentional because these systems divide themselves according to knowledge base (the store of specialized knowledge) and inference machine (which unites the procedures for fixing problems, or steps for solving them). The combination forms what is called a knowledge-based system. ***The base contains facts and rules, and the inference machine decides how to apply these rules and in which order so***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*as to obtain new knowledge.* Once the specialized knowledge is separated, it becomes easier for the designer to manipulate procedures [ROC 94a].

da Rocha at 612-13.

588. Further examples include da Rocha at 619, 620. *See generally, e.g.,* da Rocha at Sections 3, 4.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

589. Da Rocha discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

Agent 6 was installed in the Darwin workstation, which is the server for the local network and the gateway for CESUP, being monitored the ie0-bus interfaces of the UFRGS network and the ie1-bus of the CESUP local network. ***Agent 6 operated in two basic modes supported by the SunNet manager (SNM) platform:*** data monitoring and event monitoring.

For data monitoring, the agent remained in continuous execution, making a consultation at each time interval. At the end of each consultation, the data recovered was sent to the SNM manager. This periodic sending of data permits the generation of graphs (see figure 1) by the SNM platform, as well as the storage of recovered data from previous consultation by the same agent, which can then generate statistics about the status of the communication system. ***Since the SNM platform supports the automatic generation of events like “send mail”*** and permits the specification of various types of thresholds, the implementation of this kind of operation by Agent 6 brought no burden to its development. To the contrary, this facility was used in sending alerts to the network administrator.

da Rocha at 614-15.

590. Further examples include da Rocha at Section 3.

591. To the extent this limitation is not expressly disclosed by da Rocha, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to achieve fault management in a distributed networking system and to improve network efficiency.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

592. Da Rocha discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

***If the machine in which the agent is running is generating high traffic and the rest of the machines on the network are generating almost nothing, then the number of collisions will be low, not reflecting the true volume of transmission achieved.*** The true volume of traffic can only be measured by counting all of the packets that pass through the network. Meanwhile, this verified difference will not adversely affect the average desired congestion, since if the station at which the agent is running is transmitting too much, it does not perceive congestion. And for the rest of the stations, there will only be congestion when they try to transmit more, which will also be reflected at the agent's station by an increase in the number of collisions. To obtain a percentage of collisions, the relation between the number of collisions and the number of attempts at transmission which occurred during the interval between the present time and the most recent consultation [ROC 94].

da Rocha at 614.

593. Further examples include da Rocha at Fig. 4. *See generally, e.g.,* da Rocha at Section 3.

594. To the extent this limitation is not expressly disclosed by da Rocha, it would have been obvious to a person of ordinary skill in the art to perform corrective actions that included load balancing operations in a distributed networking system at least to improve network efficiency and scalability.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters to known thresholds.*

595. Da Rocha discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

This work was developed in the area of Computer Network Management. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. ***The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks,*** and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

596. Further examples include da Rocha at 612, 618, Fig. 4. *See generally, e.g.,* da Rocha at Sections 2, 3, 4.

597. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

598. Da Rocha discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example:

As previously stated, ***the concept of proactive management involves the anticipation of possible problems which may occur in a computer network and their detection before they occur,*** and not merely reporting their existence. It is fundamental that abnormal network operation be observed, that symptoms be collected and that larger problems which may come to occur be diagnosed correctly, or that anomalies be registered when it is not possible to collect enough evidence which associates an event with a known problem. ***It is also necessary to maintain constant observation of the network, so that based on this knowledge enough data can be collected in order to identify what might be a symptom and relate it to a known problem.*** In an ideal situation, the LAN management tools establish a

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

framework within which devices like smart hubs can monitor network activity and place information at distributed management platforms which automatically will generate trouble ticket, operational costs and usage reports [JAN 93].

da Rocha at 611-12.

599. Further examples include da Rocha at Abstract, 611, 612-613. *See generally, e.g.,* da Rocha at Sections 1, 3.

600. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

601. Da Rocha discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

602. Further examples include da Rocha at 613, 619. *See generally, e.g.,* da Rocha at Sections 1, 2.

- b. *running at least one thread in a first runtime environment;*

603. Da Rocha discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

604. Da Rocha discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

605. Da Rocha discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

606. Da Rocha discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

607. Da Rocha discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

608. Da Rocha discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

609. Da Rocha discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

610. Da Rocha discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

611. Da Rocha discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

612. Further examples include da Rocha at 613. 619. *See generally, e.g.,* da Rocha at Sections 1, 2.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

613. Da Rocha discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13.

For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

614. Further examples include da Rocha at 613, 619. *See generally, e.g.,* da Rocha at Sections 1, 2.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

615. Da Rocha discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

616. Further examples include da Rocha at 613, 619. *See generally, e.g.,* da Rocha at Sections 1, 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *running at least one thread in a first runtime environment;*

617. Da Rocha discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

618. Da Rocha discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

- f. *detecting there is an abnormality in the monitored operational parameters;*

619. Da Rocha discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- g. *and performing a corrective action to fix any detected abnormalities;*

620. Da Rocha discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

621. Da Rocha discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

622. Da Rocha discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

623. Da Rocha discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

624. Da Rocha discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

625. Da Rocha discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises using Dijkstra's Self Stabilization Algorithm.*

626. Da Rocha discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

627. I expect to testify that da Rocha anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that da Rocha discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-6.

**L. Anticipation by and/or Obviousness in View of *Proactive Management of Software Aging*, by Castelli, et al., published in March 2001 by IBM.**

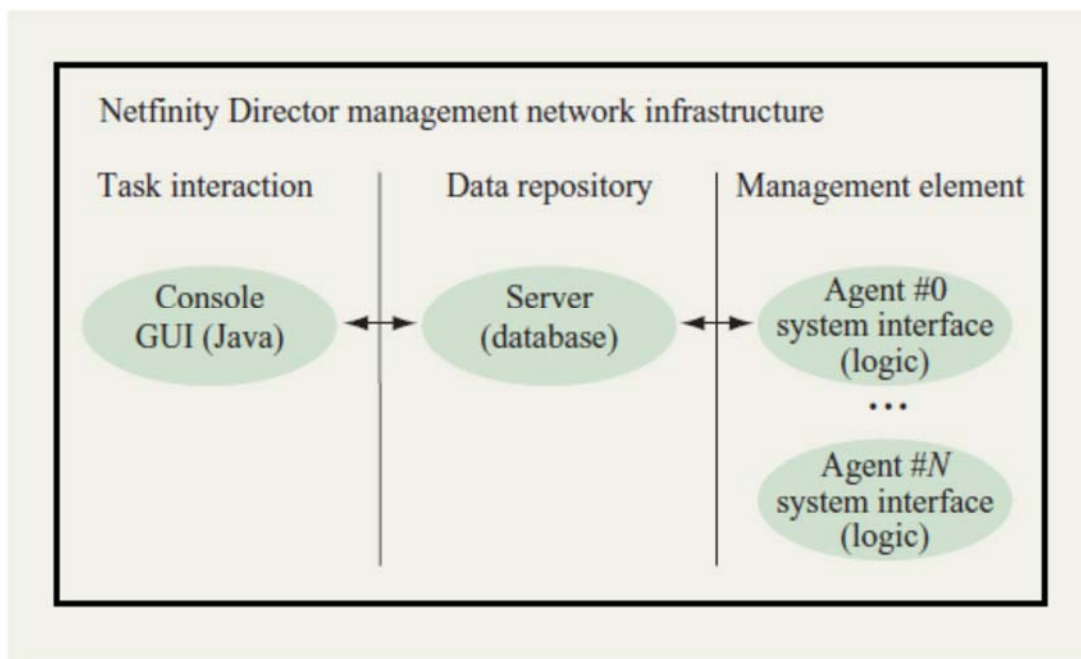
**1. *Claim 1***

**a. *A computer-implemented method, comprising:***

628. Castelli discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

Castelli at 313

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1**

IBM Director framework.

Castelli at 315; *see id.* at 312.b. *running at least one thread in a first runtime environment;*

629. Castelli discloses this element of claim 1, "running at least one thread in a first runtime environment." '659 Pat. at Cl. 1. For example:

*Management console:* The IBM Director management console is the graphical user interface (GUI) from which administrative tasks are performed. It is the primary interface with the administrator, and is used to configure the SRA as described below. ***The management console GUI is Java-based, with all state information stored on the server. It runs as a locally installed Java application in a Java Virtual Machine (JVM\*\*).***

*Management server:* The management server is the platform used for the central management server, where management databases, the server engine, and ***management application logic reside.***

*IBM Director agents:* The agents ***reside on each managed system*** (such as an xSeries server) and act as passive, ***nonintrusive native applications.*** The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

Castelli at 315; *see id.* at 317.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

630. Castelli discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

- *IBM Director agents*: The agents ***reside on each managed system (such as an xSeries server)*** and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

***In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes.*** These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:

- *Inventory*: IBM Director discovers new managed systems, collects the appropriate information about these systems, and stores it in the inventory database. It can then be viewed through either a default or a customized view.
- *Resource monitors*: Resource monitors (**Figure 2**) enable the user to view statistics and usage of critical resources on the network. Information can be collected and monitored on attributes such as CPU, disk, memory, and network. SRA is a specialized instance of a resource monitor.
- *Event management*: Event management (**Figure 3**) enables the user to view a log of events that have occurred for a managed system or group of systems and to create event action plans to associate an event with a desired action, such as sending an e-mail, starting a program, logging to a file, or invoking rejuvenation. When the SRA has detected an impending resource exhaustion, it generates events that can be viewed using this functionality.

Castelli at 315-316; *see id.* at 312, 14, 17.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

631. Castelli discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

***Prediction algorithms***

In the current version of SRA, rejuvenation can be based on elapsed time since the last rejuvenation, or on prediction of impending exhaustion.

When using timed rejuvenation, the user interface of IBM Director is used to schedule and perform rejuvenation at a period specified by the user. A

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

calendar interface allows the user to select when to rejuvenate different nodes of the cluster, and to select “blackout” times during which no rejuvenation is to be allowed.

Although this sounds rather primitive, our analysis (presented below) shows that for typical clusters that undergo aging, system availability can be improved significantly via this technique.

Castelli at 317; *see id.* at 317-18, 23.

- e. *and performing a corrective action to fix any detected abnormalities,*

632. Castelli discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

**Rejuvenation granularity**

When an exhaustion has been predicted, the question arises as to what portion of the environment should be rejuvenated. ***The simple approach to rejuvenation is to perform a node reboot.*** A single computer node is a distinct and compartmentalized unit of the user’s resource environment, and the cluster-management framework within which we are operating should handle such node failures quite adequately. The drawback is that an entire node rejuvenation may be considered too broad in scope, and the time to reboot may be significant. Perhaps only one application on that system, only one of that application’s services, or perhaps even a particular subprocess of that service is the cause of the pending exhaustion. The narrower the scope of rejuvenation, the smaller the disruption will be to the user’s business objective. ***From a functional point of view, the operating system provides various APIs to perform rejuvenation at all of these various levels.*** Whether the resource to be rejuvenated is at the system level, the application level, or even the service level, it is likely that the planned downtime (i.e., the rejuvenation) is always less disruptive than an unplanned node outage.

Because of these considerations, the Windows SRA offers two levels of rejuvenation, depending on the scope of the resource exhaustion (i.e., whether an operating system-level exhaustion or an application-level exhaustion has been identified). ***The desired level can be selected using an advanced submenu of the user interface described above.***

- *Level 1 is a service-level rejuvenation.* Generally, it can be assumed that applications written as a service are written such that a stoppage of that service will save any necessary data (both user and application) and the corresponding restart of that service will bring the application to a state of usability. In such cases, a graceful rejuvenation of that service can be performed.
- *Level 2 is an operating-system-level rejuvenation (i.e., a reboot).* The reboot performs a stop for each service on that system and then reboots the operating system. Application failover and recovery in this case are



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the responsibility of the cluster-management software, which is activated as part of the reboot process.

Castelli at 318; *see id.* at 323.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

633. Castelli discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at 317

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. ***Events provide a notification mechanism from an agent into the IBM Director environment.***

Castelli at 315; *see also* 318, 323

634. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time that an agent or its runtime environment could be programmed to request a corrective policy from an entity outside of the particular runtime environment.

wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

635. Castelli discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at 317

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. ***Events provide a notification mechanism from an agent into the IBM Director environment.***

Castelli at 315; *see also* 318, 323

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

636. Castelli discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

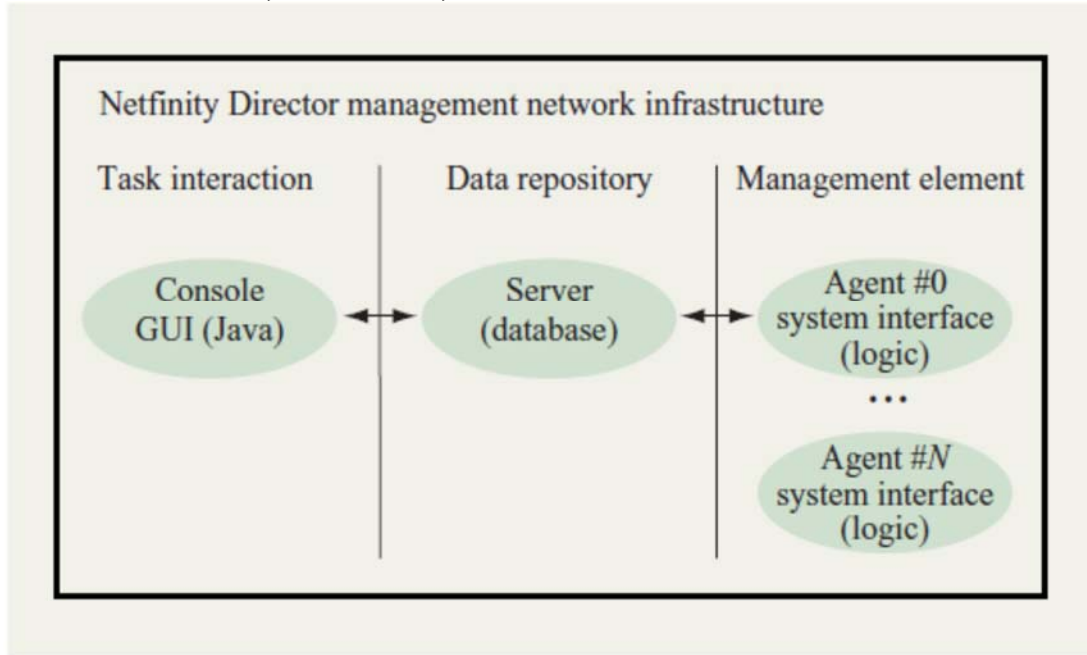
Castelli at 317

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. *Events provide a notification mechanism from an agent into the IBM Director environment.*

Castelli at 315; *see also* 318, 323.



**Figure 1**

IBM Director framework.

Castelli at 315

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

637. Castelli discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

***IP dispatching***

As part of a web-hosting framework, an IP-dispatching or load-balancing component is often present to provide scalability, availability, and load-balancing capabilities for TCP/IP applications. Early implementations

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

consisted of a domain name server (DNS) that would translate host names into IP addresses for corresponding servers, so that IP requests are routed in a round-robin fashion to a pool of servers. This approach is often called round-robin DNS; an example configuration is shown in Figure 16. Note that IP dispatching can be considered a specialized form of clustering, such that one or more nodes execute the dispatching components and the remaining nodes execute web-serving applications.

More advanced approaches are now available, such as the IBM Secureway Network Dispatcher [32]. This product provides enhanced IP-level load-balancing mechanisms and content-based routing, as well as improved management and availability functions. Figure 17 illustrates the network dispatcher operations for a LAN implementation. As part of load balancing, the dispatcher's scheduling policy is dynamically based on each server's load and availability. This is partly accomplished by having each server send periodic utilization information to the dispatcher. This utilization information can easily be augmented by health information in the form of time until resource exhaustion, degree of resource exhaustion or, in its simplest form, time remaining until a timed rejuvenation. This health information can be sent to the dispatcher in order to schedule actions for individual servers. The scheduling of these actions can also take into account aggregate loading of the web host.

Castelli at 323

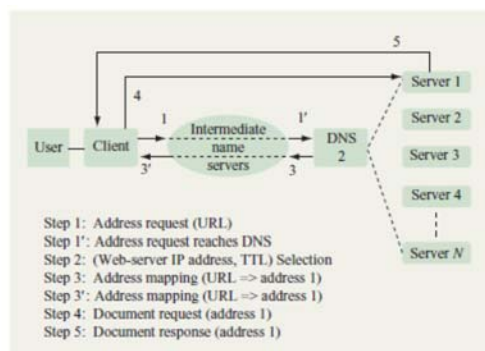


Figure 16

Example configuration of round-robin DNS.

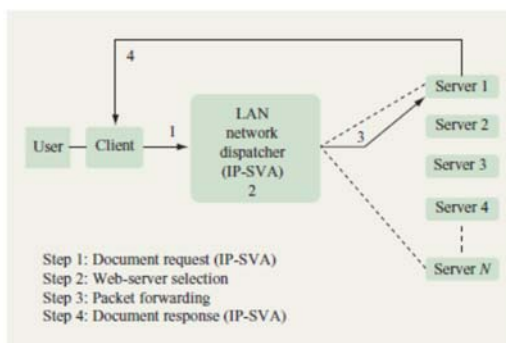


Figure 17

Network dispatcher for a LAN implementation.

Castelli at 323

### 3. *Claim 3*

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

638. Castelli discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY*****Prediction algorithms***

In the current version of SRA, rejuvenation can be based on elapsed time since the last rejuvenation, or on prediction of impending exhaustion.

When using timed rejuvenation, the user interface of IBM Director is used to schedule and perform rejuvenation at a period specified by the user. A calendar interface allows the user to select when to rejuvenate different nodes of the cluster, and to select "blackout" times during which no rejuvenation is to be allowed. Although this sounds rather primitive, our analysis (presented below) shows that for typical clusters that undergo aging, system availability can be improved significantly via this technique.

Castelli at 317

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day's worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, *see also id.* at 323

639. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**4. **Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

640. Castelli discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm."  
'659 Pat. at Cl. 6. For example:

***Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.*** The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the ***agent automatically performs the analysis.***

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and ***the analysis and extrapolation over the three-day horizon are performed using that one day's worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function*** without overfitting the data.

The ***prediction algorithm fits several types of curves to the data in the fitting window***; these curves have been selected for their ability to capture different types of temporal trends. ***A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon.*** Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below.

Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, *see also id.* 324, 27, 28, 29

641. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

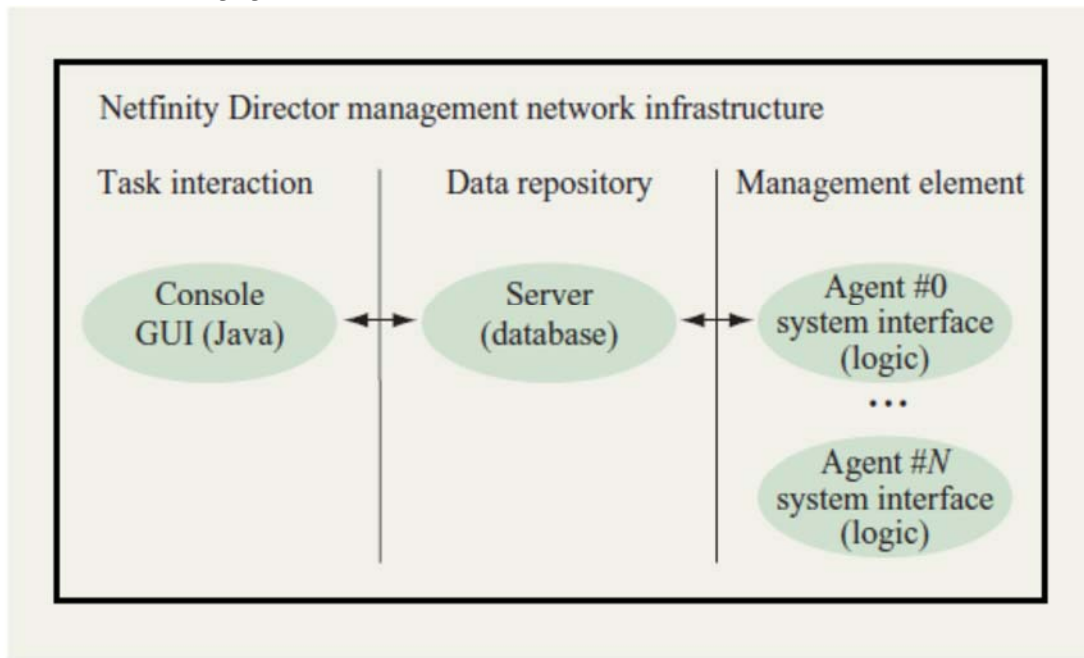
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**5. *Claim 7*

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

642. Castelli discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

Castelli at 313.



**Figure 1**

IBM Director framework.

Castelli at 315; *see id.* at 312.

- b. *running at least one thread in a first runtime environment;*

643. Castelli discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

644. Castelli discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

645. Castelli discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- e. *and performing a corrective action to fix any detected abnormalities;*

646. Castelli discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

647. Castelli discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

648. Castelli discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

649. Castelli discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

650. Castelli discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

651. Castelli discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

This technology has been incorporated into the IBM Director for xSeries servers.

Castelli at 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castelli at 323-24.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

652. Castelli discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

This technology has been incorporated into the IBM Director for xSeries servers.

Castelli at 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castelli at 323-24.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

653. Castelli discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

This technology has been incorporated into the IBM Director for xSeries servers.

Castelli at 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castelli at 323-24.

d. *running at least one thread in a first runtime environment;*

654. Castelli discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

655. Castelli discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

656. Castelli discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

657. Castelli discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

658. Castelli discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

659. Castelli discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

660. Castelli discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

661. Castelli discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

662. Castelli discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

663. Castelli discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

664. I expect to testify that Castelli anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Castelli discloses each and every limitation of those claims is included as B-12.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

665. I expect to testify that this art anticipates each Asserted Claim of the '659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as B-1.

**M. Anticipation by and/or Obviousness in View of *Network Security Management with Intelligent Agents - A First Step with SLD* ("Conti"), published in the 1998 International Workshop on Intelligent Agents for Telecommunication Applications.**

**1. *Claim 1***

**a. *A computer-implemented method, comprising:***

666. Conti discloses the preamble of claim 1, "[a] computer-implemented method, comprising . . . ." '659 Pat. at Cl. 1. For example:

In this implementation we have demonstrated that a DIANA agent system may be easily used to monitor a network and perform failure recovering. As the agent are distributed, close to the NEs they have to manage and able to report only useful information, saving bandwidth consumption.

Conti at pp. 7-8.

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Verify that the agent is detected as in state unreliable and that its domain is redistributed.

Conti at pp. 5-6, *see also id.* at 3, 4, 5.

- b. *running at least one thread in a first runtime environment;*

667. Conti discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

The agents are hierarchically organised with the possibility to share and delegate activities and responsibilities. ***Each agent has a NM domain***, which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But ***an agent may owns and manage different domains*** at the same time. In the same way an agent X may be the manager of an agent Y for an NM area, and Y be the manager of X for an other NM area. This may be explained by the fact that agents are collaborative and the manager role is a responsibility role.

Conti at p.3.

***Several internals component are articulated*** around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. ***Different kinds of skill may be developed and used***, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

668. To the extent this limitation is not expressly disclosed by Conti, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

669. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.1. A person of ordinary skill in the art would understand that multithreading would have improved the efficiency of agents in a distributed network.

- c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

670. Conti discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

The second one was a monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module use a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. ***This algorithm uses the beliefs resulting of the monitoring activity performed by the IAs, and requires to these IAs to have some common representative elements monitored*** (fig 3)

Conti at p. 5; *see also id.* at Fig. 3 (showing agents’ common representative elements monitored).

The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks ***every IA to monitor their NEs*** and report the results.

Conti at p. 5..

d. *detecting if there is an abnormality in the monitored operational parameters;*

671. Conti discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

The third developed skill module use a ***SLD (System Level Diagnosis)*** algorithm to verify the state of a distributed system. This algorithm uses the beliefs ***resulting of the monitoring activity*** performed by the IAs, and requires to these IAs to have some common representative elements monitored (fig 3). The next skill developed was ***a general Fault Management module***, which just gets the domain attributed by the Agent Management skill, and asks ***every IA to monitor their NEs and report the results.***

Conti at p. 5

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.

Conti at pp. 5-6; *see also Id.* at 2, 4, 7.

- e. *and performing a corrective action to fix any detected abnormalities,*

672. Conti discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

The next skill developed was a ***general Fault Management module***, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an Interface skill composed by the skill module itself and a java applet. Its objective is to start the Fault Management, display the monitoring results, ***force an agent to report erroneous information*** and force an NE (through the help of the monitoring) to stop any reporting.

Conti at p. 5

In this implementation we have demonstrated that a DIANA agent system may be easily used to monitor a network and ***perform failure recovering***.

Conti at p. 7.

The common IA properties are:

- Autonomy: does not require instruction, ***knows what to do in what circumstances***.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask on agent to do something for it.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Deliberative: *is able to reason according to its knowledge (beliefs).*
- Learning capacities: *acquires and uses new knowledge.*
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.
- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: *responds on event.*
- Security: knows if an agent is corrupted or not.

Conti at pp. 5-6; *see also Id.* at 2, 4, 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

673. Conti discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

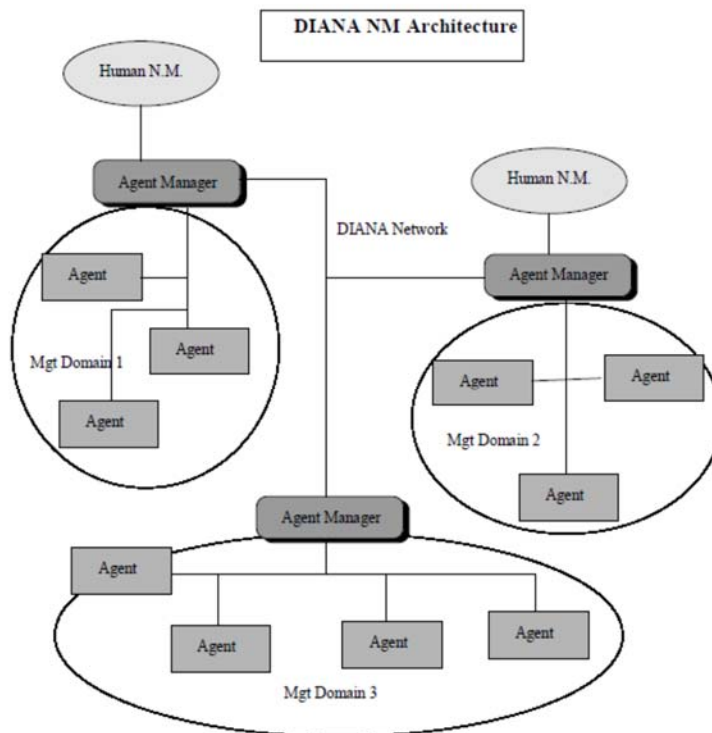
To perform their tasks the agents need skills, which they may load from an other agent when they need them. The agents may be *addressed by the users* using a web browser if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at p. 4

Conti at Fig. 1 (showing DIANA NM Architecture).



**Figure 1**

The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also ***it requires the agents to start the monitoring of their new NEs***. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

674. Conti discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Interface skill composed by the skill module itself and a java applet. Its objective is to start the Fault Management, display the monitoring results, force an agent to report erroneous information and force an NE (through the help of the monitoring) to stop any reporting.

Conti at p. 5.

The common IA properties are:

- Autonomy: does not require instruction, *knows what to do in what circumstances*.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask an agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.
- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: responds on event.
- Security: knows if an agent is corrupted or not.

Conti at p. 2.

At this moment the SLD diagnoses that an agent is unreliable and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master's Brain to the Agent Management skill *which understand that it has to redistribute the agent domain (NEs) of this agent to the other available agents*. The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

675. Conti discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an Interface skill composed by the skill module itself and a java applet. Its objective is to start the Fault Management, display the monitoring results, force an agent to report erroneous information and force an NE (through the help of the monitoring) to stop any reporting.

Conti at p. 5.

The common IA properties are:

- Autonomy: does not require instruction, *knows what to do in what circumstances.*
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask on agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.
- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: responds on event.
- Security: knows if an agent is corrupted or not.

Conti at p. 2.

At this moment the SLD diagnoses that an agent is unreliable and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master’s Brain to the Agent Management skill *which understand that it has to redistribute the agent domain (NEs) of this agent to the other available agents.* The Fault Management skill has

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

676. Conti discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

In this implementation we have demonstrated that a DIANA agent system may be easily used to monitor a network and ***perform failure recovering. As the agent are distributed***, close to the NEs they have to manage and able to report only useful information, saving bandwidth consumption.

Conti at p. 7)

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and ***that its domain is redistributed.***

Conti at pp. 5-6; *see also id.* at 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

677. To the extent this limitation is not expressly disclosed by Conti, it would have been obvious to a person of ordinary skill in the art in view of Buchanan.<sup>2</sup> Conti discloses improving efficiency and scalability in a distributed network with agents having one or more associated underlying resources. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

678. Conti discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

Several internal components are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain loads a skill, it gets the services the skill offers, the beliefs it uses or creates, ***and the prerequisite other skill*** it needs to execute. So when a new belief has been created, the brain knows and forwards it to all interested skill modules.

Conti at p. 4.

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agents (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previously designed master agent.

---

<sup>2</sup> WJ Buchanan, M Naylor and AV Scott, Enhancing Network Management using Mobile Agents (published January 10, 1997) (“Buchanan”).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at p. 5.

679. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. **Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

680. Conti discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example:

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. ***The idea was to realise a management system able to continue its work even if one of its elements fails.*** The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. ***The third developed skill module use a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This algorithm uses the beliefs resulting of the monitoring activity performed by the IAs, and requires to these IAs to have some common representative elements monitored*** (fig 3). The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an Interface skill composed by the skill module itself and a java applet. Its objective is to start the Fault Management, display the monitoring results, force an agent to report erroneous information and force an NE (through the help of the monitoring) to stop any reporting.

Conti at p. 5.

At this moment ***the SLD diagnoses that an agent is unreliable*** and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master's Brain to the Agent Management skill ***which understand that it has to redistribute the agent domain (NEs)*** of this agent to the other available agents. The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at p. 7.

681. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

682. Conti discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. For example:

Several internal components are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain loads a skill, it gets the services the skill offers, the beliefs it uses or creates, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain knows and forwards it to all interested skill modules.

Conti at p. 4.

The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wished frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module uses a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This algorithm uses the beliefs resulting from the monitoring activity performed by the IAs, and requires these IAs to have some common representative elements monitored (fig 3). The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an Interface skill composed by the skill module itself and a Java applet.

Conti at p. 5.

The architecture we propose is based on the extensible/flexible capabilities of IA. The agents are hierarchically organised with the possibility to share and delegate activities and responsibilities. Each agent has a NM domain,



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But an agent may own and manage different domains at the same time.

Conti at p. 3.

b. *running at least one thread in a first runtime environment;*

683. Conti discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

684. Conti discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

685. Conti discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

686. Conti discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

687. Conti discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

688. Conti discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

689. Conti discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

690. Conti discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

691. Conti discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

In this implementation we have demonstrated that ***a DIANA agent system*** may be easily used to monitor a network and perform failure recovering. As the agent are distributed, close to the NEs they have to manage and able to report only useful information, saving bandwidth consumption.

Conti at pp. 7-8.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. The idea was to *realise a management system* able to Continue its work even if one of its elements fails. The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP.

Conti at p. 5.

The last part is devoted to a case study involving agents managed from a web browser interface to explain the collaboration between agents. For this purpose, we use a scenario of fault detection, which explains the use of a *System Level Diagnosis* skill module. This example introduces also how several skill modules may be used to reach a same goal.

Conti at p. 1.

The second one was *a* monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module use a SLD (System Level Diagnosis) algorithm *to verify the state of a distributed system*.

Conti at p. 5.

b. *a processor;*

692. Conti discloses this element of claim 13, "a processor." '659 Pat. at Cl. 13. For example:

Several internal component are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*system to perform a method comprising:*

693. Conti discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

Several internal components are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain loads a skill, it gets the services the skill offers, the beliefs it uses or creates, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain knows and forwards it to all interested skill modules.

Conti at p. 4.

694. To the extent this limitation is not expressly disclosed by Conti, it would have been obvious to a person of ordinary skill in the art, because it was well known to use memory coupled to the processor to store instructions to be executed by the processor.

d. *running at least one thread in a first runtime environment;*

695. Conti discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

696. Conti discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

697. Conti discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

698. Conti discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

699. Conti discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

700. Conti discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

701. Conti discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*load balancing operation.*

702. Conti discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

703. Conti discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

704. Conti discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

705. I expect to testify that Conti anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Conti discloses each and every limitation of those claims is included as Ex. B-17.

706. I expect to testify that this art anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

**N. Anticipation by and/or Obviousness in View of *Network Security Management with Intelligent Agents* (“NSMIA”), NOMS 2000. 2000 IEEE/IFIP Network Operations and Management Symposium 'The Networked Planet:**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Management Beyond 2000.**1. ***Claim 1***a. *A computer-implemented method, comprising:*

707. NSMIA discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

***Moreover, multi-agent systems, as a sub-domain of DAI, are viewed as computational systems*** in which several autonomous and intelligent agents interact and work together in order to perform a set of tasks and to satisfy a set of goals [1][5].

NSMIA at 7.

The aim of this paper is to propose ***a multi-agent system to model the network*** security management, particularly the network intrusion detection.

NSMIA at 2.

In this paper, we suggest to ***improve network security management*** by integrating DAI approach based ***on multi-agent system technique in Intrusion Detection Systems (IDS)***. We propose a new approach based on ***providing the NSM (Network Security Management) hosts with additional functionalities***. These entities become more intelligent, capable of making various decisions with autonomy to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

NSMIA at 2.

Intrusion detection is a practical approach for enhancing ***the security of computer and network systems***. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

- the behavior-based intrusion detection approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;
- the knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database.

The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions.

NSMIA at 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In our architecture, we propose a hierarchical structure of autonomous agents. In the functional architecture (see Figure 1), we distinguish two: a *Manager Layer* and a *Local Layer*.

NSMIA at 8.

***Multi-Agent Systems technology can be useful for efficiently designing and maintaining secure networks.*** Indeed, networks evolve at a rapid pace in terms of the number and type of components and user access queries as well as intrusion possibilities. Features such as autonomy, adaptability and flexibility of the “intelligent” agent paradigm allow managing network evolution in a controlled way. The focus of our work concerns one critical security management issue that is *intrusion detection*. We propose a novel approach called IA-NSM (Intelligent Agents for Network Security Management) for intrusion detection using intelligent agent technology. IA-NSM provides a flexible integration of a multi-agent system in a classical networked environment to enhance its protection level against inherent attacks.

NSMIA at 1.

The key characteristics of our architecture include autonomy, adaptability, efficiency and distribution to make the network intrusion detection more flexible and less costly in term of maintenance. In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. ***It is based on a multi-agent system architecture (see Figure 1). It is viewed as a collection of autonomous and intelligent agents located in specific network entities named NSM hosts.*** These agents cooperate and communicate in order to perform intrusion detection tasks efficiently and achieve consequently better performance.

NSMIA at 8.

b. *running at least one thread in a first runtime environment;*

708. NSMIA discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

Our agents are composed of seven modules: perception, deliberation, communication, action, interface, report; each executing a different task. The supervisor entity coordinates tasks of the different modules.

- ***A perception module: that gathers all security-relevant events produced in the agent environment.***
- A communication module: that allows agents to communicate their analysis, decisions and knowledge.
- An action module: its role is to take appropriate actions when an intruder is detected.
- A report generation: establishes reports on detected attacks to be sent to the administrator.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- A deliberation module: that enables agent intelligence and autonomy. The hybrid agent should be able to reason and extrapolate by relying on built knowledge and experience in a rational way. *Decisions of the agent depend on the security environment status, the neighboring system evolution and its mental attitudes.*"

NSMIA at 10-11.

- **Autonomy:** is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal state *and/or external environment*.
- **Socialability:** *is the capability of an agent to integrate itself in a large environment* populated by a society of agents with which the agent has to exchange messages to achieve purposeful actions. This property is satisfied even when systems have to share their knowledge and mental attitudes (beliefs, goals, desires, etc.).
- **Proactivity:** is the ability of an agent to anticipate situations and change its course of action. It is a relevant property which occurs in network and system management in order to avoid disastrous effects on global performance. Indeed, proactive agents are capable of exhibiting goal-direct behaviors by taking some initiatives [6].
- **Reactivity:** this kind of behavior means that *the agent reacts in real-time to changes* that occur in its *environment*.
- **Adaptability:** is the ability of an agent to modify its behavior over time to fulfill its problem-solving goals.

NSMIA at 7.

An agent is a pro-active [sic] entity. It does not simply act in response to the received messages from the other agents. For example, *it interacts with its environment* and deliberates to determine the most appropriate action.

NSMIA at 11.

Ferber [5] defines an agent as a computational or physical entity situated in an environment (either real or virtual) which is able to act in the environment, to perceive and partially to represent its environment and to communicate with other agents.

NSMIA at 7.

- The Manager Layer manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: Security Policy Manager Agent, Extranet Manager Agent and Intranet Manager Agent.
  - The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control *Intranet Manager Agents* (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The Extranet Manager Agent communicates with the Security Policy Manager Agent. This latter can specify new security policy, new monitoring tasks or new attacks to detect. The EMA is also responsible for distributing the set of Local Agents to each IMA.
- The Intranet Manager Agent (IMA) manages the security of a local network. It controls the *Local Agents* and analyzes the monitored events reported by these agents.

NSMIA at 9.

709. To the extent this limitation is not expressly disclosed by NSMIA, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

710. To the extent this limitation is not met, it would have been obvious in view of Turek '070. *See* Ex. B-1 at limitation 1.1. A person of ordinary skill in the art would understand that multithreading would have improved the efficiency of agents in a distributed network.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

711. NSMIA discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat. at Cl.

1. For example:

- The Manager Layer manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: Security Policy Manager Agent, Extranet Manager Agent and Intranet Manager Agent.
  - The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.
  - The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control Intranet Manager Agents (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The *Extranet Manager Agent*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

communicates with the *Security Policy Manager Agent*. ***This latter can specify new security policy, new monitoring tasks or new attacks to detect.*** The *EMA* is also responsible for distributing the set of *Local Agents* to each IMA.

- The Intranet Manager Agent (IMA) manages the security of a local network. It controls the *Local Agents* and analyzes the monitored events reported by these agents.

NSMIA at 9.

The *manager layer* interacts with the *local layer* by sending goals, delegating specific ***monitoring***/detection tasks and receiving pertinent reports and alarms. In each level, agents communicate and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.

NSMIA at 9.

According to the kind of data, an IDS uses to detect intrusive activity, we distinguish three types of IDS:

- ***Host-based IDS***, which are designed to ***monitor*** a single host. They use their own host Operating System's audit trail as the main source of input for detecting intrusions.
- ***Distributed Host-based IDS***, which are in charge of ***monitoring several hosts***. They perform intrusion detection using Operating System's audit trail or information from multiple monitored hosts. This information is processed on a central site.

NSMIA at 4-5.

- The *Local Layer* manages the security of a domain, which is constituted by a set of hosts. This layer is composed of a group of *Local agents*, which have specific functions. In fact, ***the Manager Layer specifies to the Local Layer the activities that must be monitored.*** These activities can be classified in *Extranet*, *Intranet* and *Local* activities."

NSMIA at 9.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

712. NSMIA discloses this element of claim 1, "detecting if there is an abnormality in the monitored operational parameters." '659 Pat. at Cl. 1. For example:

We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, capable of making various decisions with autonomy ***to detect intrusions*** and to overcome their bad effects.

NSMIA at 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The deliberation module represents beliefs, intentions and knowledge of the agent. It is responsible 1) for generating adequate responses to the messages transmitted by the communication module, or to the changes detected by the perception module, and 2) for achieving the agent goal(s). ***This goal can be the detection of a specific attack.***

NSMIA at 12.

***Intrusion detection*** is a practical approach for enhancing the security of computer and network systems. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

- the ***behavior-based intrusion detection*** approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;
- the ***knowledge-based intrusion detection*** approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database.

NSMIA at 4.

- Proactivity: is the ability of ***an agent to anticipate situations and change its course of action.*** It is a relevant property which occurs in network and system management in order to avoid disastrous effects on global performance. Indeed, proactive agents are capable of exhibiting goal-direct behaviors by taking some initiatives [6].
- Reactivity: this kind of behavior means that ***the agent reacts in real-time to changes*** that occur in its environment.
- Adaptability: is the ability of an agent to modify its behavior over time to fulfill its problem-solving goals.

NSMIA at 7.

- The ***Manager Layer manages the global security*** of a network. This network can be local or distributed. In this layer we identify three levels of agents: Security Policy Manager Agent, Extranet Manager Agent and Intranet Manager Agent.
  - The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.
  - The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control ***Intranet Manager Agents*** (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to ***perform another analysis on suspicious events in order to confirm or not the detection of an attack.*** It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The Extranet Manager Agent communicates with the Security Policy Manager Agent. This latter can specify new security policy, new monitoring tasks ***or new***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*attacks to detect.* The EMA is also responsible for distributing the set of Local Agents to each IMA.

- The *Intranet Manager Agent (IMA) manages the security of a local network.* It controls the Local Agents and analyzes the monitored events reported by these agents.

NSMIA at 9.

Administrators do not have to concern about all the security problems. They interact with the agent from a high level using security policies. Security policies tell the agents what behavior they should exhibit when attacks occur. Hence, communications between agents permit to collect information. This information *permits the agents to identify attacks* that can not be detected if it is static. Giving more autonomy to the agent permits the system to react in “real time” to attacks and to take necessary actions to avoid severe consequences of the attack.

NSMIA at 8.

The key characteristics of our architecture include autonomy, adaptability, efficiency and distribution to make the network *intrusion detection* more flexible and less costly in term of maintenance. In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. It is based on a multi-agent system architecture (see Figure 1). It is viewed as a collection of autonomous and intelligent agents located in specific network entities named NSM hosts. These agents cooperate and communicate in order *to perform intrusion detection tasks* efficiently and achieve consequently better performance.

NSMIA at 8.

- e. *and performing a corrective action to fix any detected abnormalities,*

713. NSMIA discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, capable of making various decisions with autonomy *to detect intrusions and to overcome their bad effects.*

NSMIA at 2.

Administrators do not have to concern about all the security problems. They interact with the agent from a high level using security policies. Security policies tell the agents what behavior they should exhibit when attacks occur. Hence, communications between agents permit to collect information. This information permits the agents to identify attacks that can not be detected if it is static. Giving more autonomy to the agent permits the system to react in “real time” to attacks and *to take necessary actions to avoid severe consequences of the attack.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

NSMIA at 8.

- An *action module*: *its role is to take appropriate actions when an intruder is detected.*
- A report generation: establishes reports on detected attacks to be sent to the administrator....
- The deliberation module represents beliefs, intentions and knowledge of the agent. It is responsible 1) for *generating adequate responses to the messages* transmitted by the communication module, or to the changes detected by the perception module, and 2) for achieving the agent goal(s). This goal can be the detection of a specific attack.

NSMIA at 11-12.

- Proactivity: is the ability of *an agent to anticipate situations and change its course of action*. It is a relevant property which occurs in network and system management *in order to avoid disastrous effects on global performance*. Indeed, proactive agents are capable of exhibiting goal-direct behaviors by taking some initiatives [6].
- Reactivity: this kind of behavior means that *the agent reacts in real-time to changes* that occur in its environment.
- Adaptability: is the ability of an agent to modify its behavior over time to fulfill its problem-solving goals.

NSMIA at 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

714. NSMIA discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

- The communication module manages the interactions between the agent and the other agents of its group(s), no matter what machine they are running on. It defines the mailbox of the agent and the way the messages are received and enqueued for later interpretation. An agent may need some others information to refine its analysis. In this case, *it asks other agents to give it the necessary information.*

NSMIA at 12.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- An interface module: interacts with the security administrator receiving administrator requests/specifications, delivering reports, sending alarms when an attack is detected and ***asking for additional information or confirmation when necessary***. For example, the administrator can ask for the current network security status. This module exists only in the SPMA.

NSMIA at 11.

Administrators do not have to concern about all the security problems. They interact with the agent from a high level using security policies. ***Security policies tell the agents what behavior they should exhibit when attacks occur***. Hence, communications between agents permit to collect information. This information permits the agents to identify attacks that can not be detected if it is static. Giving more autonomy to the agent permits the system to react in “real time” to attacks and to take necessary actions to avoid severe consequences of the attack.

NSMIA at 8.

The DAI (Distributed Artificial Intelligence) concept [1] consists of a group of individual named agents that have distributed environments. Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with ***the information coming from neighboring agents permits the agent to make the best (optimum in some sense) decision***.

NSMIA at 2.

- The Manager Layer manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: Security Policy Manager Agent, Extranet Manager Agent and Intranet Manager Agent.
  - The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.
  - The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control Intranet Manager Agents (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. ***It can also ask for another data processing and delegate then new monitoring tasks to the IMAs***. The Extranet Manager Agent communicates with the Security Policy Manager Agent. This latter can specify new security policy, new monitoring tasks or new attacks to detect. The EMA is also responsible for distributing the set of Local Agents to each IMA.
  - The Intranet Manager Agent (IMA) manages the security of a local network. It controls the Local Agents and analyzes the monitored events reported by these agents.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- The Local Layer manages the security of a domain, which is constituted by a set of hosts. This layer is composed of a group of Local agents, which have specific functions. In fact, the Manager Layer specifies to the Local Layer the activities that must be monitored. These activities can be classified in Extranet, Intranet and Local activities. According to this classification, we distinguish 3 kinds of Local Agents: Extranet Local Agent, Intranet Local Agent and Internal Local Agent.

NSMIA at 9.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

715. NSMIA discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

- Proactivity: is the ability of *an agent to anticipate situations and change its course of action*. It is a relevant property which occurs in network and system management *in order to avoid disastrous effects on global performance*. Indeed, proactive agents are capable of exhibiting goal-direct behaviors by taking some initiatives [6].
- Reactivity: this kind of behavior means that *the agent reacts in real-time to changes* that occur in its environment.
- Adaptability: is the ability of an agent to modify its behavior over time to fulfill its problem-solving goals.

NSMIA at 7.

- An *action module: its role is to take appropriate actions when an intruder is detected*.
- A report generation: establishes reports on detected attacks to be sent to the administrator....
- The deliberation module represents beliefs, intentions and knowledge of the agent. It is responsible 1) for *generating adequate responses to the messages* transmitted by the communication module, or to the changes detected by the perception module, and 2) for achieving the agent goal(s). This goal can be the detection of a specific attack.

NSMIA at 11-12.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

716. NSMIA discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

- The Manager Layer manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: Security Policy Manager Agent, Extranet Manager Agent and Intranet Manager Agent.
  - The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.
  - The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control Intranet Manager Agents (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The Extranet Manager Agent communicates with the Security Policy Manager Agent. This latter can specify new security policy, new monitoring tasks or new attacks to detect. The EMA is also responsible for distributing the set of Local Agents to each IMA.
  - The Intranet Manager Agent (IMA) manages the security of a local network. It controls the Local Agents and analyzes the monitored events reported by these agents.
- The Local Layer manages the security of a domain, which is constituted by a set of hosts. This layer is composed of a group of Local agents, which have specific functions. In fact, the Manager Layer specifies to the Local Layer the activities that must be monitored. These activities can be classified in Extranet, Intranet and Local activities. According to this classification, we distinguish 3 kinds of Local Agents: Extranet Local Agent, Intranet Local Agent and Internal Local Agent.

NSMIA at 9.

The *manager layer* interacts with the *local layer* **by sending goals**, delegating specific monitoring/detection tasks and receiving pertinent reports and alarms. In each level, agents communicate and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.

NSMIA at 9.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

717. To the extent this limitation is not expressly disclosed by NSMIA, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

718. NSMIA discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

- Distribution of activities: This important aspect is provided by most existing approaches. It is very important to distribute the control of security management among a number of entities that can monitor the network access at different points.

NSMIA at 6.

The DAI (Distributed Artificial Intelligence) concept [1] consists of a group of individual named agents that have distributed environments. Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with the information coming from neighboring agents permits the agent to make the best (optimum in some sense) decision.

NSMIA at 2.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

719. NSMIA discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

Intrusion detection is a practical approach for enhancing the security of computer and network systems. The goal of IDS is to detect attacks

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

especially in real-time fashion. These systems use one or both approaches of intrusion detection:

- the behavior-based intrusion detection approach, which discovers intrusive activity by *comparing the user or system behavior with a normal behavior profile*;
- the knowledge-based intrusion detection approach, which detects intrusions upon a comparison between *parameters of the user's session and known pattern attacks stored in a database*.

NSMIA at 4.

- A deliberation module: that enables agent intelligence and autonomy. The hybrid agent should be able to reason and *extrapolate by relying on built knowledge* and experience in a rational way. Decisions of the agent depend on the security environment status, the neighboring system evolution and its mental attitudes.

NSMIA at 11.

720. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

721. NSMIA discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example:

Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with the information coming from neighboring agents permits the agent to make the best (optimum in some sense) decision. In this paper, we suggest to improve network security management by integrating DAI approach based on multi-agent system technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. *These entities become more intelligent, capable of making various decisions with autonomy* to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming "intelligent".

NSMIA at 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Intelligence: the term “Intelligence” means that the agent is able to exhibit a certain level of intelligence priority, ranging from predefined actions (planning) up to self learning (define new actions)

NSMIA at 7.

722. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

723. NSMIA discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

***Moreover, multi-agent systems, as a sub-domain of DAI, are viewed as computational systems*** in which several autonomous and intelligent agents interact and work together in order to perform a set of tasks and to satisfy a set of goals [1][5].

NSMIA at 7.

The aim of this paper is to propose ***a multi-agent system to model the network*** security management, particularly the network intrusion detection.

NSMIA at 2.

In this paper, we suggest to ***improve network security management*** by integrating DAI approach based ***on multi-agent system technique in Intrusion Detection Systems (IDS)***. We propose a new approach based on ***providing the NSM (Network Security Management) hosts with additional functionalities***. These entities become more intelligent, capable of making various decisions with autonomy to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

NSMIA at 2.

Intrusion detection is a practical approach for enhancing ***the security of computer and network systems***. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- the behavior-based intrusion detection approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;
- the knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database.

The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions.

NSMIA at 4.

In our architecture, we propose a hierarchical structure of autonomous agents. In the functional architecture (see Figure 1), we distinguish two: a *Manager Layer* and a *Local Layer*.

NSMIA at 8.

***Multi-Agent Systems technology can be useful for efficiently designing and maintaining secure networks.*** Indeed, networks evolve at a rapid pace in terms of the number and type of components and user access queries as well as intrusion possibilities. Features such as autonomy, adaptability and flexibility of the “intelligent” agent paradigm allow managing network evolution in a controlled way. The focus of our work concerns one critical security management issue that is *intrusion detection*. We propose a novel approach called IA-NSM (Intelligent Agents for Network Security Management) for intrusion detection using intelligent agent technology. IA-NSM provides a flexible integration of a multi-agent system in a classical networked environment to enhance its protection level against inherent attacks.

NSMIA at 1.

The key characteristics of our architecture include autonomy, adaptability, efficiency and distribution to make the network intrusion detection more flexible and less costly in term of maintenance. In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. ***It is based on a multi-agent system architecture (see Figure 1). It is viewed as a collection of autonomous and intelligent agents located in specific network entities named NSM hosts.*** These agents cooperate and communicate in order to perform intrusion detection tasks efficiently and achieve consequently better performance.

NSMIA at 8.

- b. *running at least one thread in a first runtime environment;*

724. NSMIA discloses this element of claim 7, “running at least one thread in a first runtime environment.” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

725. NSMIA discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

726. NSMIA discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

727. NSMIA discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

728. NSMIA discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

729. NSMIA discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

730. NSMIA discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

731. NSMIA discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

732. NSMIA discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

We propose a novel approach called IA-NSM (Intelligent Agents for Network Security Management) for intrusion detection using intelligent agent technology. IA-NSM provides a flexible integration of ***a multi-agent system*** in a classical networked environment to enhance its protection level against inherent attacks.

NSMIA at 1.

The aim of this paper is to propose ***a multi-agent system*** to model the network security management, particularly the network intrusion detection.

NSMIA at 2.

In this paper, we suggest to improve network security management by integrating DAI approach based ***on multi-agent system*** technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, capable of making

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

various decisions with autonomy to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

NSMIA at 2.

The key characteristics of our architecture include autonomy, adaptability, efficiency and distribution to make the network intrusion detection more flexible and less costly in term of maintenance. In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. It is based on *a multi-agent system architecture* (see Figure 1). It is viewed as a collection of autonomous and intelligent agents located in specific network entities named NSM hosts. These agents cooperate and communicate in order to perform intrusion detection tasks efficiently and achieve consequently better performance.

NSMIA at p. 8)

Intrusion detection is a practical approach for enhancing *the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

- the behavior-based intrusion detection approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;
- the knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database.

The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions.

NSMIA at 4.

Moreover, *multi-agent systems*, as a sub-domain of DAI, are viewed as *computational systems* in which several autonomous and intelligent agents interact and work together in order to perform a set of tasks and to satisfy a set of goals [1][5].

NSMIA at 7.

The DIMA architecture (see Figure 3) relies on two layers:

A first layer made up of interactive modules, which represent the different concurrent agent behaviors such as communicating, reasoning and perceiving. They provide the agents with some properties described in [6] such as autonomy and cooperation. For example, the communication module manages the interaction between the agent and some other agents of the system. Therefore, it is very important to make the agent cooperative.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

NSMIA at 12.

b. *a processor;*

733. NSMIA discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

In DIMA, an ATN is associated to each module. In the ATN, we distinguish two main states: an initialization state "INIT" and a supervision state "WAIT" that *manages the processing of each module*.

In this paper, we provide an ATN for an *Extranet Local Agent* dedicated to the detection of both doorknob rattling and IP spoofing attacks.

NSMIA at 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

734. NSMIA discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

We propose a novel approach called IA-NSM (Intelligent Agents for Network Security Management) for intrusion detection using intelligent agent technology. IA-NSM provides a flexible integration of *a multi-agent system* in a classical networked environment to enhance its protection level against inherent attacks.

NSMIA at 1.

The aim of this paper is to propose *a multi-agent system* to model the network security management, particularly the network intrusion detection.

NSMIA at 2.

In this paper, we suggest to improve network security management by integrating DAI approach based *on multi-agent system* technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, capable of making various decisions with autonomy to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

NSMIA at 2.

The key characteristics of our architecture include autonomy, adaptability, efficiency and distribution to make the network intrusion detection more



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

flexible and less costly in term of maintenance. In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. It is based on *a multi-agent system architecture* (see Figure 1). It is viewed as a collection of autonomous and intelligent agents located in specific network entities named NSM hosts. These agents cooperate and communicate in order to perform intrusion detection tasks efficiently and achieve consequently better performance.

NSMIA at 8.

Intrusion detection is a practical approach for enhancing *the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

- the behavior-based intrusion detection approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;
- the knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database.

The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions.

NSMIA at 4.

Moreover, *multi-agent systems*, as a sub-domain of DAI, are viewed as *computational systems* in which several autonomous and intelligent agents interact and work together in order to perform a set of tasks and to satisfy a set of goals [1][5].

NSMIA at 7.

The DIMA architecture (see Figure 3) relies on two layers:

A first layer made up of interactive modules, which represent the different concurrent agent behaviors such as communicating, reasoning and perceiving. They provide the agents with some properties described in [6] such as autonomy and cooperation. For example, the communication module manages the interaction between the agent and some other agents of *the system*. Therefore, it is very important to make the agent cooperative.

NSMIA at 12.

735. To the extent this limitation is not expressly disclosed by NSMIA, it would have been obvious to a person of ordinary skill in the art, because it was well known to use a memory coupled to the processor to store instructions to be executed by the processor.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *running at least one thread in a first runtime environment;*

736. NSMIA discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

737. NSMIA discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

738. NSMIA discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

739. NSMIA discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

740. NSMIA discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

741. NSMIA discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

742. NSMIA discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

743. NSMIA discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

744. NSMIA discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises using Dijkstra's Self Stabilization Algorithm.*

745. NSMIA discloses claim 18, "[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm," at least for all the reasons explained above regarding claim 6.

746. I expect to testify that NSMIA anticipates each Asserted Claim of the '659 Patent. A claim chart illustrating that [NSMIA] discloses each and every limitation of those claims is included as Ex. B-19.

747. I expect to testify that this art anticipates each Asserted Claim of the '659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. 1.

**O. Anticipation by and/or Obviousness in View of *INCA: An Agent-based Network Control Architecture*, published in the 1998 International Workshop on Intelligent Agents for Telecommunication Applications.**

748. As explained in detail below and in the chart attached as Ex. B-13, INCA anticipates and renders obvious the claims of the '659 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

**1. Claim 1**

**a. A computer-implemented method, comprising:**

749. INCA discloses the preamble of claim 1, "[a] computer-implemented method, comprising . . . ." '659 Pat. at Cl. 1. For example:

This paper describes the design and implementation of INCA, **an open architecture for the distributed management of multi-service networks and systems applications**. The Intelligent Network Control Architecture is populated by stationary and mobile intelligent agents. These agents perform monitoring and control of network and systems components, thereby supporting the integrated management of networks and services.

INCA at Abstract.

750. Further examples include INCA at Sections 3, 4.1, 5. *See generally, e.g.,* INCA at Section 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

751. INCA discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

In order to demonstrate the usefulness of agent prioritization for network management applications, we chose a simple scenario. It consists of a set of connected LANs. At each LAN there is one network element hosting an INCA station. ***Each INCA station provides a set of managed objects allowing agents to access the network elements in the LAN.*** Furthermore, we consider a management station launching agents for monitoring and control. Monitoring agents are stationary and monitor dedicated network elements of a LAN. Thus, there typically are more than one monitoring agents at a station. Different to these continuously running monitoring agents, there are control agents which are launched interactively by a network operator to perform a usually short task. For these agents instantaneous task completion is desired.

INCA at 12.

752. Further examples include INCA at Sections 4.1.1, 4.5, 4.6; INCA at Fig. 2. *See generally, e.g.,* INCA at Section 4.

753. To the extent this limitation is not expressly disclosed by INCA, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

754. INCA discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

**Concurrent execution of multiple agents.** Multiple agents - each with its own thread of control - can be executed concurrently. The agents are scheduled according to their priority attribute by an INCA specific scheduling scheme, ***which for example takes into account the current system load.*** No agent of lower priority will be executed if there is an agent with higher priority in a "runnable" state present at a station.” INCA at Section 4.1.1)

- *Network Control Agents* support the network manager e.g. in configuration and fault management of the network. The tasks they pursue are demand driven, perhaps triggered by network faults, and often have to be carried out as fast as possible.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- *Service Management Agents* support the service provider, e.g. by a service subscription agent. They use the network environment together with the platform services to install and establish end user services, or reconfigure them.
- *Network Maintenance Agents perform repeatedly occurring management tasks, e.g. the gathering of data from selected network elements or fault analysis.* Their tasks are carried out repeatedly, only rarely requiring adjustments by the network management instance.
- *Network Monitoring Agents perform routine tasks such as the filtering of raw data collected from network elements.* Often, such agents are stationary, since their task is to monitor a concrete element of the network, or a link.

INCA at 6.

755. Further examples include INCA at Sections 4.4, 4.6. *See generally, e.g.,* INCA at Section 4.

756. To the extent this limitation is not expressly disclosed by INCA, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

757. INCA discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

Suppose that a mobile maintenance agent is used to perform a routine task at all the network elements, or a fixed subset of it. ***An example could be an agent which examines the network elements and checks the current memory usage and the disk space used.*** In case of an alarming situation at the network element, it will send a message to the network manager, otherwise it proceeds along its itinerary.

INCA at 8.

758. Further examples include INCA at Sections 4.2, 4.6. *See generally, e.g.,* INCA at Section 4.

- e. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities,*

759. INCA discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

4.1.1 Local Services

*A station is local to a network element and provides the following services:*

**Concurrent execution of multiple agents. Multiple agents – each with its own thread of control – can be executed concurrently.** The agents are scheduled according to their priority attribute by an INCA specific scheduling scheme, which for example takes into account the current system load. ***No agent of lower priority will be executed if there is an agent with higher priority in a "runnable" state present at a station.***

INCA at 4-5.

760. Further examples include INCA at Sections 4.2, 4.4. *See generally, e.g.,* INCA at Section 4.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

761. INCA discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

**Loading of agent code. Agent code can be transferred between stations.** Three schemes for code transfer are supported: push, pull, and migrate. Pushed code is sent from an agent code repository to a dedicated station, pulled code is loaded by the station from a repository, and in migrating code is sent from one (non-repository) station to another. All stations support the same code format, i.e. the same code can be executed at any stations. In order to improve performance, caching of agent code is used.

INCA at 5.

762. Further examples include INCA at Sections 4.1.2, 4.5; INCA at Fig. 2. *See generally, e.g.,* INCA at Section 4.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

763. INCA discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

4.1.1 Local Services

***A station is local to a network element and provides the following services:***

***Concurrent execution of multiple agents. Multiple agents – each with its own thread of control – can be executed concurrently.*** The agents are scheduled according to their priority attribute by an INCA specific scheduling scheme, which for example takes into account the current system load. No agent of lower priority will be executed if there is an agent with higher priority in a "runnable" state present at a station.

INCA at 4-5.

764. Further examples include INCA at Sections 4.1.1, 4.1.2, 4.2, 4.5; INCA at Fig. 2. See generally, e.g., INCA at Section 4.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

765. INCA discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

4.1.1 Local Services

***A station is local to a network element and provides the following services:***

***Concurrent execution of multiple agents. Multiple agents – each with its own thread of control – can be executed concurrently.*** The agents are scheduled according to their priority attribute by an INCA specific scheduling scheme, which for example takes into account the current system load. No agent of lower priority will be executed if there is an agent with higher priority in a 'runnable' state present at a station.

INCA at 4-5.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

766. Further examples include INCA at Section 4.1.2. *See generally, e.g.,* INCA at Section 4.

767. To the extent this limitation is not expressly disclosed by INCA, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

768. INCA discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

**Concurrent execution of multiple agents.** Multiple agents - each with its own thread of control - can be executed concurrently. *The agents are scheduled according to their priority attribute by an INCA specific scheduling scheme, which for example takes into account the current system load.* No agent of lower priority will be executed if there is an agent with higher priority in a "runnable" state present at a station.

INCA at 5.

769. Further examples include INCA at Sections 4.2, 4.4, 4.6. *See generally, e.g.,* INCA at Section 4.

770. To the extent this limitation is not expressly disclosed by INCA, it would have been obvious to a person of ordinary skill in the art to perform corrective actions that included load balancing operations in a distributed networking system at least to improve network efficiency and scalability.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters to known thresholds.*

771. INCA discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

- *Repetitive tasks can be avoided if software agents learn from experience.*
- *Fault tolerance can be increased through the autonomy and learning capability of management agents.*

INCA at 3.

772. Further examples include INCA at Sections 4.1.1, 4.2, 4.4. *See generally, e.g.,* INCA at Section 4.

773. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

774. INCA discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example:

**Loading of agent code. Agent code can be transferred between stations.** Three schemes for code transfer are supported: push, pull, and migrate. Pushed code is sent from an agent code repository to a dedicated station, pulled code is loaded by the station from a repository, and in migrating code is sent from one (non-repository) station to another. All stations support the same code format, i.e. the same code can be executed at any stations. In order to improve performance, caching of agent code is used.

INCA at 5.

775. Further examples include INCA at Sections 3, 4.2, 5, 7. *See generally, e.g.,* INCA at Section 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

776. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

777. INCA discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

Network management today is typically based on a client-server model with SNMP [2] and CMIP [16] as the de facto ***standards for monitoring and for managing a network of computers and devices***. Managed objects, like hosts, routers, bridges, switches, printers etc., provide read/write access to a set of variables through a management process. A management station can then communicate with the management processes in a client-server relationship.

INCA at 3.

778. Further examples include INCA at Sections 4.1, 4.1.1, 4.1.2, 5. *See generally, e.g.,* INCA at Section 4.

- b. *running at least one thread in a first runtime environment;*

779. INCA discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

780. INCA discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters;*

781. INCA discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

782. INCA discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

783. INCA discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

784. INCA discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

a. *The computer-readable medium of claim 7, wherein the corrective*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

785. INCA discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

786. INCA discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

787. INCA discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

Network management today is typically based on a client-server model with SNMP [2] and CMIP [16] as the de facto ***standards for monitoring and for managing a network of computers and devices***. Managed objects, like hosts, routers, bridges, switches, printers etc., provide read/write access to a set of variables through a management process. A management station can then communicate with the management processes in a client-server relationship.

INCA at 3.

788. Further examples include INCA at Sections 4.1.1, 4.1.2, 5. *See generally, e.g.,* INCA at Section 4.

- b. *a processor;*

789. INCA discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

Network management today is typically based on a client-server model with SNMP [2] and CMIP [16] as the de facto ***standards for monitoring and for managing a network of computers and devices***. Managed objects, like hosts, routers, bridges, switches, printers etc., provide read/write access to a set of variables through a management process. A management station can

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

then communicate with the management processes in a client-server relationship.

INCA at 3.

790. Further examples include INCA at Sections 4.1, 4.1.1, 4.1.2, 5. *See generally, e.g.,* INCA at Section 4.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

791. INCA discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

Network management today is typically based on a client-server model with SNMP [2] and CMIP [16] as the de facto ***standards for monitoring and for managing a network of computers and devices***. Managed objects, like hosts, routers, bridges, switches, printers etc., provide read/write access to a set of variables through a management process. A management station can then communicate with the management processes in a client-server relationship.

INCA at 3.

792. Further examples include INCA at Sections 4.1, 4.1.1, 4.1.2, 5. *See generally, e.g.,* INCA at Section 4.

- d. *running at least one thread in a first runtime environment;*

793. INCA discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

794. INCA discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

- f. *detecting there is an abnormality in the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

795. INCA discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

796. INCA discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

797. INCA discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

798. INCA discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

799. INCA discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

800. INCA discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

801. INCA discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

802. INCA discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

803. I expect to testify that INCA anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that INCA discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-13.

804. I expect to testify that this art anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

**P. Anticipation by and/or Obviousness in View of *Distributed Network Security Management Using Intelligent Agents*, by Boudaoud, et al., published in April**



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****1998 at the HP Openview University Association Workshop.**1. ***Claim 1***a. *A computer-implemented method, comprising:*

805. Boudaoud discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

Intrusion detection is a practical approach *for enhancing the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. *We focus our work on network intrusion detection systems and we present below two specific systems DIDS* (Distributed Intrusion Detection System) and **CSM** (Co-operating Security Managers).

Boudaoud at 2.

806. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 2, 3.

b. *running at least one thread in a first runtime environment;*

807. Boudaoud discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

The Management Agent Execution Environment (MAEE) is a set of components necessary for the execution and the migration of [intelligent agents].

Boudaoud at 7

808. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

809. To the extent this limitation is not expressly disclosed by Boudaoud, this Claim Limitation would have been obvious to a person of ordinary skill in the art, because multithreaded agent architectures were well known at the time.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

810. Boudaoud discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

Intrusion detection is a practical approach *for enhancing the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. *We focus our work on network intrusion detection systems and we present below two specific systems DIDS (Distributed Intrusion Detection System) and CSM (Co-operating Security Managers).*

Boudaoud at 2.

811. Further examples include Boudaoud at Sec. 5.1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

812. To the extent this limitation is not expressly disclosed by Boudaoud, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

813. Boudaoud discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

Intrusion detection is a practical approach for enhancing the security of computer and network systems. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, *which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. *We focus our work on network intrusion detection systems and we present below two specific systems DIDS (Distributed Intrusion Detection System) and CSM (Co-operating Security Managers).*

Boudaoud at 2.

814. Further examples include Boudaoud at Secs. 5.1, 5.2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

- e. *and performing a corrective action to fix any detected abnormalities,*

815. Boudaoud discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

Nevertheless, different types of agents reflect a set of properties, which common among them and are described below [12]:

- Autonomy: is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal and/or external environments
- Co-operation: an Agent is co-operative and is able to have a social ability. *This sociability allows an agent to interact with other agents for the purpose of performing tasks that are beyond the capability of a particular agent.* This capability goes from delegation (distribution of sub-tasks) to peer-to-peer interworking.

Boudaoud at 4.

**Skills** can be downloaded dynamically into the agent inside its Skill Base. *The main role of the Skill Manager is to check the availability of pre-requisite skills required by newly loaded skills and if they are not yet loaded, it must search for them either locally or on distant agents.* It is also responsible for disposing off no useful skills to keep the agent's size as small as possible. During its operation, the skill can update or delete existing beliefs or create new ones. A skill operation may depend on beliefs created by other skills, and the Skill Manager is therefore in charge of dispatching asynchronously these beliefs to the interested skill in a transparent way. It holds all the necessary information about the skills in the Skill Base.

Boudaoud at 5.

816. Further examples include Boudaoud at Sec. 5.2; Boudaoud at Fig. 1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

817. Boudaoud discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example, when an agent determines that it does not have one of the skills necessary for its task, it can request the skill from another agent that has it:

Nevertheless, different types of agents reflect a set of properties, which common among them and are described below [12]:

- Autonomy: is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal and/or external environments
- Co-operation: an Agent is co-operative and is able to have a social ability. ***This sociability allows an agent to interact with other agents for the purpose of performing tasks that are beyond the capability of a particular agent.*** This capability goes from delegation (distribution of sub-tasks) to peer-to-peer interworking.

Boudaoud at 4.

818. Further examples include Boudaoud at Secs. 3.1, 5.2; Boudaoud at Fig. 1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

819. Boudaoud discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

**Skills** can be downloaded dynamically into the agent inside its Skill Base. ***The main role of the Skill Manager is to check the availability of pre-requisite skills required by newly loaded skills and if they are not yet loaded, it must search for them either locally or on distant agents.*** It is also responsible for disposing off no useful skills to keep the agent’s size as small as possible. During its operation, the skill can update or delete existing beliefs or create new ones. A skill operation may depend on beliefs created

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by other skills, and the Skill Manager is therefore in charge of dispatching asynchronously these beliefs to the interested skill in a transparent way. It holds all the necessary information about the skills in the Skill Base.

Boudaoud at 5.

820. Further examples include Boudaoud at Secs. 5.1, 5.2; Boudaoud at Fig. 1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

821. Boudaoud discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

Nevertheless, different types of agents reflect a set of properties, which common among them and are described below [12]:

- Autonomy: is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal and/or external environments
- Co-operation: an Agent is co-operative and is able to have a social ability. This sociability allows an agent to interact with other agents for the purpose of performing tasks that are beyond the capability of a particular agent. ***This capability goes from delegation (distribution of sub-tasks) to peer-to-peer interworking.***

Boudaoud at 4.

822. Further examples include Boudaoud at Secs. 3.1, 5.1, 5.2; Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

823. To the extent this limitation is not expressly disclosed by Boudaoud, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time an entity outside of a particular runtime environment could be used to monitor events in that and other subordinate runtime environments, each runtime environment including multiple threads in order to achieve fault management in a distributed networking system and to improve network efficiency.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**2. *Claim 2*

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

824. Boudaoud discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

Intrusion detection is a practical approach for enhancing the security of computer and network systems. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, *which detects well-known intrusions*. We focus our work on network intrusion detection systems and we present below two specific systems **DIDS** (Distributed Intrusion Detection System) and **CSM** (Co-operating Security Managers).

Boudaoud at 2.

825. To the extent this limitation is not expressly disclosed by Boudaoud, it would have been obvious to a person of ordinary skill in the art to perform corrective actions that included load balancing operations in a distributed networking system at least to improve network efficiency and scalability.

3. *Claim 3*

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

826. Boudaoud discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

Intrusion detection is a practical approach for enhancing the security of computer and network systems. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

use one or both of two approaches of intrusion detection. *The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database.* The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. We focus our work on network intrusion detection systems and we present below two specific systems **DIDS** (Distributed Intrusion Detection System) and **CSM** (Co-operating Security Managers).

Boudaoud at 2.

827. Further examples include Boudaoud at Sections 3, 5.

828. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. **Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

829. Boudaoud discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm."

'659 Pat. at Cl. 6. For example:

The [Intelligent Agents (IAs)] should monitor the network in order to detect security-relevant events and then react according to the behaviour specified by the administrator. The IAs may also report the administrator Workstation the security-relevant events. In case of a special event, the IAs may also co-operate to check or have some information in order to have a more precise status on the special event. For example, if an agent detects an "unauthorizedAccessAttempt", it can co-operate with others agents to check if there are other login attempts on their hosts. An example of this functionality is given below. Suppose that an intruder came from an external network, in the night or in the weekend, obtained an access, and had an unauthorised activity. The agent that is monitoring all the incoming connections detects an "unknownAddress" and an "outOfHoursActivity" event. This agent can track the intruder by migrating to the host were the intruder is working. If the intruder "travel" from one host to another host, migrating agent can follows intruder's activities by co-operating with the others agents, responsible for monitoring these hosts. If one of the co-operating agents detects, for instance an "unauthorizedAccessAttempt", or an "suspiciousActivitiy", *the first agent can migrate to the host on the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*entry of the internal network and close the connection or to ask another agent to do it.*

Boudaoud at 8.

830. Further examples include Boudaoud at Secs. 2, 5.3, 5.4. *See generally, e.g.,* Boudaoud at Sections 3, 5.

831. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

832. Boudaoud discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

Intrusion detection is a practical approach ***for enhancing the security of computer and network systems***. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. ***We focus our work on network intrusion detection systems and we present below two specific systems DIDS*** (Distributed Intrusion Detection System) and ***CSM*** (Co-operating Security Managers).

Boudaoud at 2.

833. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 2, 3.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *running at least one thread in a first runtime environment;*

834. Boudaoud discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

835. Boudaoud discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

836. Boudaoud discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- e. *and performing a corrective action to fix any detected abnormalities;*

837. Boudaoud discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

838. Boudaoud discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

839. Boudaoud discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

840. Boudaoud discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

841. Boudaoud discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

842. Boudaoud discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

Intrusion detection is a practical approach ***for enhancing the security of computer and network systems***. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. *We focus our work on network intrusion detection systems and we present below two specific systems DIDS* (Distributed Intrusion Detection System) and **CSM** (Co-operating Security Managers).

Boudaoud at 2.

843. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 2, 3.

b. *a processor;*

844. Boudaoud discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

Intrusion detection is a practical approach *for enhancing the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. *We focus our work on network intrusion detection systems and we present below two specific systems DIDS* (Distributed Intrusion Detection System) and **CSM** (Co-operating Security Managers).

Boudaoud at 2.

845. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 2, 3.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

846. Boudaoud discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

Intrusion detection is a practical approach *for enhancing the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. ***We focus our work on network intrusion detection systems and we present below two specific systems DIDS (Distributed Intrusion Detection System) and CSM (Co-operating Security Managers).***

Boudaoud at 2.

847. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 2, 3.

d. *running at least one thread in a first runtime environment;*

848. Boudaoud discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

849. Boudaoud discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

850. Boudaoud discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

851. Boudaoud discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

852. Boudaoud discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

853. Boudaoud discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

854. Boudaoud discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

855. Boudaoud discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

856. Boudaoud discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

857. Boudaoud discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

858. I expect to testify that Boudaoud anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Boudaoud discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-14.

**Q. Anticipation by and/or Obviousness in View of *Distributed Policy Based Management Enabling Policy Adaptation on Monitoring Using Active Network Technology*, by Yoshihara, et al., published in October 2001 at the 12th International Workshop on Distributed Systems.**

859. As explained in detail below and in the chart attached as Ex. B-21, Yoshihara anticipates and renders obvious the claims of the ’659 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

860. Yoshihara discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

In this paper, in the context of IETF policy-based management, we propose a new ***policy-based management scheme enabling the policy life-cycle management***. In the scheme, we introduce a new management script describing not only policies but also how their life-cycle should be managed. ***The script is executed on the managed system*** assumed to have enough computation resources, ***as well as in the active network technology***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*[Pso99, RS00], the distributed management paradigm such as MbD (Management by Delegation) [YGY91, GY98], and the management by mobile software agents [ZHG99, GGGO00].* By executing the script on the managed system, the scheme can make the current policy-based management more scalable by reducing management traffic, more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden.

Yoshihara at 2.

861. Further examples include Yoshihara at 1, 4; Yoshihara at Fig. 1.

b. *running at least one thread in a first runtime environment;*

862. Yoshihara discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

2. For the purpose of scalability and reliability, we execute the management script on the managed system having enough computational resources, as well as in the active network technology [Pso99, RS00], the distributed management paradigm such as MbD (Management by Delegation) [YGY91, GY98], and the management by mobile software agents [ZHG99, GGGO00]. If we cannot expect such a managed system, the management script should be executed on a management system or a surrogate host. The surrogate host provides enough resources to execute the management script on behalf of a managed system and a facility to communicate with the managed system in a standardized or proprietary manner.

Yoshihara at 4-5.

863. Further examples include Yoshihara at 2, 5, 6, 7, 8, 9, 11; Yoshihara at Figs. 1, 9, 11.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

864. Yoshihara discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) **traffic monitoring** to see if the enforced policy works at operators' will, and 3) policy adaptation (updating the condition and action clauses) based on the monitoring, as shown in Fig. 4.

Yoshihara at 4.

865. Further examples include Yoshihara at 5, 6, 7, 9; Yoshihara at Fig. 9.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *detecting if there is an abnormality in the monitored operational parameters;*

866. Yoshihara discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

*When a threshold violation occurs*, the management script notifies the management system of the event (Fig. 7(a)(3)). The management system may use this event as a trigger for a policy adaptation.

Yoshihara at 5.

867. Further examples include Yoshihara at 7, 9; Yoshihara at Tables 1, 2.

- e. *and performing a corrective action to fix any detected abnormalities,*

868. Yoshihara discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

The purpose of Diffserv is to establish peak boundaries on traffic values and the boundaries are given in the static forms. So, in order to make the most of limited network resources, a ***mechanism for adapting and optimizing the boundaries*** according to the ***dynamic nature of network utilization*** is required. Such a requirement is also stated in some early literatures [Wie94, Goh98]. In the policy-based management, therefore, it is inevitable to manage policy life-cycle management. We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) traffic monitoring to see if the enforced policy works at operators' will, and 3) ***policy adaptation (updating the condition and action clauses)*** based on the monitoring, as shown in Fig. 4.

Yoshihara at 4.

869. Further examples include Yoshihara at 4, 6, 7; Yoshihara at Figs. 4, 8-9.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

870. Yoshihara discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

For the purpose of scalability and reliability, ***we execute the management script on the managed system*** having enough computational resources, as



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

well as in the active network technology [Pso99, RS00], the distributed management paradigm such as MbD (Management by Delegation) [YGY91, GY98], and the management by mobile software agents [ZHG99, GGG000]. *If we cannot expect such a managed system, the management script should be executed on a management system or a surrogate host.* The surrogate host provides enough resources to execute the management script on behalf of a managed system and a facility to communicate with the managed system in a standardized or proprietary manner.

Yoshihara at 4-5.

871. Further examples include Yoshihara at 1, 2, 3, 5, 7, 9, 11; Yoshihara at Figs. 2-4,  
11.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

872. Yoshihara discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

In the scheme, we provide a new management script describing *policies* and also how their life-cycle should be managed, and *execute the script on the managed systems called active nodes*. The scheme can make the current policy-based management more scalable by reducing management traffic, more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden.

Yoshihara at 1.

873. Further examples include Yoshihara at 2, 3, 5, 7; Yoshihara at Figs. 2, 3.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

874. Yoshihara discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

For the purpose of *monitoring* to see if the enforced policies actually work as intended, items to be monitored associated with the policies, monitoring intervals, and thresholds for events are provided in the management script.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Each management script monitors the items with the given monitoring intervals (Fig.7(a)(2)).

Yoshihara at 5.

875. Further examples include Yoshihara at 4-5, 6-7; Yoshihara at Figs. 8, 9, 11.

2. **Claim 2**

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

876. Yoshihara discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

**2.2 Overview of Differentiated Services**

*Diffserv (Differentiated Services)* [BBC+98] is one of the promising policy enforcement mechanisms standardized by IETF. Figure 2 shows an overview of Diffserv. Figure 3 gives examples of policies for Diffserv that we consider throughout the paper.

At each *ingress edge interface*, *ingress traffic is classified by means of MF (Multi-Field) classifier* specifying one or more key-value pairs in a packet, such as source/destination IP addresses and source/destination port numbers. *For each classified traffic, a profile providing a rule for determining whether a particular packet is in-profile or out-of-profile, or how the packet should be prioritized or controlled, is applied. For example, a profile in the form of simple token bucket may specify a peak rate and a peak burst size.* Depending on the result, *the packet is marked* with a 6-bit DSCP (Diffserv Code Point), *which represents a forwarding treatment of the packet* in the Diffserv network, in the IPv4 Type of Service octet or the IPv6 Traffic Class octet. *Some out-of-profile packets may be discarded* without marking.

At each core interface, the traffic is classified by means of BA (Behavior Aggregate) classifier specifying a DSCP. For each classified traffic, as well as at the ingress edge interface, a profile is applied. Some packets are remarked with another DSCP and others may be discarded depending on the result.

The policy A in Fig. 3 classifies the packets with their Source IP Address (SrcIP) 10.0.0.0/24 from others. Each packet in in-profile is marked with the DSCP “101110” representing such a forwarding treatment that the packet is sent with no loss and less delay, called EF (Expedited Forwarding). Other packets in out-of-profile is marked with the DSCP “000000” representing best-effort forwarding treatment.

Yoshihara at 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

877. Further examples include Yoshihara at 7-10; Yoshihara at Figs. 2, 3; Yoshihara at Table 1.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

878. Yoshihara discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

*When a threshold violation occurs*, the management script notifies the management system of the event (Fig. 7(a)(3)). The management system may use this event as a trigger for a policy adaptation.

Yoshihara at 5.

879. Further examples include Yoshihara at 7, 9; Yoshihara at Tables 1, 2.

880. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

881. Yoshihara discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example:

A number of ***adaptation algorithms***, from simple ones to complex ones, may be possible. Below, we show three simple but practical adaptation algorithms each for the item to be adapted in Section 4.3.1, while an investigation of more suitable one is out of the scope of the paper and is left as a future work.

Yoshihara at 7-8.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

882. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

883. Yoshihara discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. For example:

A network operator, first, decides a policy to be enforced on a managed system such as a router and a switch (Fig.1(1)).

Yoshihara at 2.

884. Further examples include Yoshihara at 9, 11; Yoshihara at Figs. 1, 11.

- b. *running at least one thread in a first runtime environment;*

885. Yoshihara discloses this element of claim 7, "running at least one thread in a first runtime environment," at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

886. Yoshihara discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

887. Yoshihara discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d.

- e. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

888. Yoshihara discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

889. Yoshihara discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

890. Yoshihara discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**6. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

891. Yoshihara discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

**7. Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

892. Yoshihara discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. **Claim 13**

a. *A system, comprising:*

893. Yoshihara discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

In this paper, in the context of IETF policy-based management, we propose a new ***policy-based management scheme enabling the policy life-cycle management***. In the scheme, we introduce a new management script describing not only policies but also how their life-cycle should be managed. ***The script is executed on the managed system*** assumed to have enough computation resources, ***as well as in the active network technology [Pso99, RS00], the distributed management paradigm such as MbD (Management by Delegation) [YGY91, GY98], and the management by mobile software agents [ZHG99, GGG000]***. By executing the script on the managed system, the scheme can make the current policy-based management more scalable by reducing management traffic, more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden.

Yoshihara at 2.

894. Further examples include Yoshihara at 1, 4; Yoshihara at Fig. 1.

b. *a processor;*

895. Yoshihara discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

When a managed system receives the request (Fig. 10(4)), the ***execution environment in the managed system*** optionally loads the necessary class files (Fig. 10(5)), and ***instantiates*** the management script. The management script then sets the provided policies to be enforced to the diffserv module (Fig. 10(6)), and responses to the management system if the enforcement has completed successfully or not (Fig. 10(7)).

Yoshihara at 9.

896. Further examples include Yoshihara at 2, 11; Yoshihara at Figs. 1, 11.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*system to perform a method comprising:*

897. Yoshihara discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

When a managed system receives the request (Fig. 10(4)), the ***execution environment in the managed system*** optionally loads the necessary class files (Fig. 10(5)), and ***instantiates*** the management script. The management script then sets the provided policies to be enforced to the diffserv module (Fig. 10(6)), and responses to the management system if the enforcement has completed successfully or not (Fig. 10(7)).

Yoshihara at 9.

898. Further examples include Yoshihara at 2, 11; Yoshihara at Figs. 1, 11.

d. *running at least one thread in a first runtime environment;*

899. Yoshihara discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

900. Yoshihara discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

901. Yoshihara discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

902. Yoshihara discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

903. Yoshihara discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

904. Yoshihara discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

905. Yoshihara discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

906. Yoshihara discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises comparing the monitored operational parameters to known thresholds.*

907. Yoshihara discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

908. Yoshihara discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

909. I expect to testify that Yoshihara anticipates and/or renders obvious each Asserted Claim of the ’659 Patent. A claim chart illustrating that Yoshihara discloses and/or renders obvious each and every limitation of those claims is included as Ex. B-21.

**R. Anticipation by and/or Obviousness in View of Cisco NetRanger, published by Wheelgroup/Cisco by August 1997 at the latest.**

1. ***Claim 1***

a. *A computer-implemented method, comprising:*

910. Cisco NetRanger discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

NetRanger is an intrusion detection system that can be plugged into your TCP/IP networks in a variety of ways. It detects and responds to unauthorized activity in real time. Unauthorized activity includes attempts to circumvent an existing firewall, router, and network security policies, as well as misuse of authorized services. NetRanger is an enterprise-wide solution that can monitor a large number of networks via centralized management. As shown in Figure 1.1, NetRanger consists of three basic components: a Sensor, a communication system, and a Director.

2.1.1 User’s Guide<sup>3</sup> at 1-1; *see also id* at 1-2.

911. Further examples are also found at

---

<sup>3</sup> NetRanger User’s Guide, Version 2.1.1 (Cisco Systems, Inc., 1998) (“2.1.1 User’s Guide”)

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

912. NetRanger discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

The Sensor handles real-time intrusion detection and device management. It uses a rules-based engine to distill large volumes of IP network traffic into meaningful events. A Sensor can either capture network traffic directly from the network or receive it from a network device, such as an STK packet filter firewall.

The Packet Filtering Device is a router or firewall residing on the network at a point of entry to other networks. The Sensor can reconfigure these devices on the fly to shun the source of an attack. Many of these devices can also serve as the data source for the intrusion Detection subsystem. Device management is optional because there are places within networks where detection is all that is required—e.g., at connections between internal networks.

2.1.1 User’s Guide at 1-4; *see also id.* at 4-2, 4-50.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

913. NetRanger discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

The Sensor handles real-time intrusion detection and device management. It uses a rules-based engine to distill large volumes of IP network traffic into meaningful events. A Sensor can either capture network traffic directly from the network or receive it from a network device, such as an STK packet filter firewall.

The Packet Filtering Device is a router or firewall residing on the network at a point of entry to other networks. The Sensor can reconfigure these devices on the fly to shun the source of an attack. Many of these devices can also serve as the data source for the intrusion Detection subsystem. Device management is optional because there are places within networks where detection is all that is required—e.g., at connections between internal networks.

2.1.1 User’s Guide at 1-4; *see also id.* at 1-5, 1-6, 1-7, 1-13, 4-95.

d. *detecting if there is an abnormality in the monitored operational parameters;*

914. NetRanger discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The Sensor handles real-time intrusion detection and device management. It uses a rules-based engine to distill large volumes of IP network traffic into meaningful events. A Sensor can either capture network traffic directly from the network or receive it from a network device, such as an STK packet filter firewall.

The Packet Filtering Device is a router or firewall residing on the network at a point of entry to other networks. The Sensor can reconfigure these devices on the fly to shun the source of an attack. Many of these devices can also serve as the data source for the intrusion Detection subsystem. Device management is optional because there are places within networks where detection is all that is required-e.g., at connections between internal networks.

2.1.1 User's Guide at 1-4; *see also id.* at 1-5, 1-6.

- e. *and performing a corrective action to fix any detected abnormalities,*

915. NetRanger discloses this element of claim 1, "and performing a corrective action to fix any detected abnormalities." '659 Pat. at Cl. 1. For example:

The Director displays events, monitors Sensors, and analyzes data. It communicates with one or more Sensor systems via the Communication System. The Director contains two basic subsystems: the Security Management Interface (SMI) and the Data Management Package (DMP).

The SMI's collection of graphical interfaces allows display of security alarms, management of remote Sensors and network devices, and response to attacks. The SMI integrates with network management applications, such as HP OpenView® and IBM NetView®.

The DMP analyzes data. The Director DMP is a set of data analysis tools that analyzes Sensor data independently of SMI activities. The DMP consists of three components: data collection, data management, and data analysis. Although all components can be easily integrated into an existing Director system, all data should be exported onto a separate database server. In this way, the SMI and DMP components can be configured, secured, and tuned independently.

2.1.1 User's Guide at 1-5.

As noted earlier, the NetRanger Director consists of two major subsystems:

- Security Management Interface (SMI)
- Data Management Package (DMP)

These two subsystems provide centralized command and control of an organization's Sensors. From a capabilities perspective, these two subsystems manage and monitor the Sensors, collect and analyze data, and facilitate user operation. All of these capabilities are supported by the smid, configd, loggerd, sapd, and eventd daemons diagrammed in Figure 1.3 .

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

There is also an application called `nrdirmap` that serves as the interface between `smid` and HP Open View or IBM NetView.

2.1.1 User's Guide at 1-13; *see also id.* at 1-7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

916. NetRanger discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

**Distribution Hierarchies**

Another feature that complements alternate routing is the ability to build hierarchies of Sensor and Director systems through the use of message propagation. Instead of broadcasting events from a Sensor onto multiple hosts, information can be sent to a single host, which can then propagate packets onto other platforms defined in its local configuration files. Figure 1.6 illustrates this concept via a simple hierarchy of Director machines.

In addition to providing performance benefits and fault tolerance, distribution hierarchies can simplify system management. For example, local Director machines might be responsible for monitoring from 9AM to 5PM and then transfer control onto a central Director every evening.

2.1.1 User's Guide at 1-11; *see also id.* at 1-12.

wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,

917. NetRanger discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

**Shunning**

Shunning an attack involves reconfiguration and reloading of a network device's security filters. This type of automated response should only be configured for attack signatures with a low probability of false positive detection. For example, a SATAN attack is less ambiguous than running VRFY off Mail Port 25. A shun can also be executed from the Director system in response to an attack or suspicious activity.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Shunning requires careful review before it is deployed, whether as a set of automatic rules or operational guidelines for staff. In case an alarm is generated from a critical host or network, NetRanger can be configured to never shun that host or network. This safety mechanism will prevent denial of service attacks using the NetRanger infrastructure.

2.1.1 User's Guide at 1-9; *see also id.* at 2-3, 2-4, 2-6.

wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.

918. NetRanger discloses this element of claim 1, "wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads." '659 Pat. at Cl. 1. For example:

#### Distribution Hierarchies

Another feature that complements alternate routing is the ability to build hierarchies of Sensor and Director systems through the use of message propagation. Instead of broadcasting events from a Sensor onto multiple hosts, information can be sent to a single host, which can then propagate packets onto other platforms defined in its local configuration files. Figure 1.6 illustrates this concept via a simple hierarchy of Director machines.

In addition to providing performance benefits and fault tolerance, distribution hierarchies can simplify system management. For example, local Director machines might be responsible for monitoring from 9AM to 5PM and then transfer control onto a central Director every evening.

2.1.1 User's Guide at 1-11; *see also id.* at 1-4, 1-12, 4-50, 4-2.

## 2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

919. NetRanger discloses claim 2, "[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation." '659 Pat. at Cl. 2. For example, Cisco NetRanger's corrective action comprises a load balancing action because shun policies are configured to perform load balancing operations on the device's main CPU.

#### Shunning

Shunning an attack involves reconfiguration and reloading of a network device's security filters. This type of automated response should only be

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

configured for attack signatures with a low probability of false positive detection. For example, a SATAN attack is less ambiguous than running VRFY off Mail Port 25. A shun can also be executed from the Director system in response to an attack or suspicious activity.

Shunning requires careful review before it is deployed, whether as a set of automatic rules or operational guidelines for staff. In case an alarm is generated from a critical host or network, NetRanger can be configured to never shun that host or network. This safety mechanism will prevent denial of service attacks using the NetRanger infrastructure.

#### 2.1.1 User's Guide at 1-9.

Additionally, to the extent this limitation is not expressly disclosed, this limitation is rendered obvious by at least Cantrill or Buchanan<sup>4</sup>. It would have been obvious at the time of invention to combine NetRanger with Cantrill or Buchanan to perform load balancing by routing traffic to one or more network devices based on thread utilization. The motivation of this modification would be to efficiently manage network capacity, or prevent a device under attack from being temporarily overwhelmed where traffic cannot be shunned.

#### Cantrill

1) *Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer. (Cantrill at p. 253, sec. 1).

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

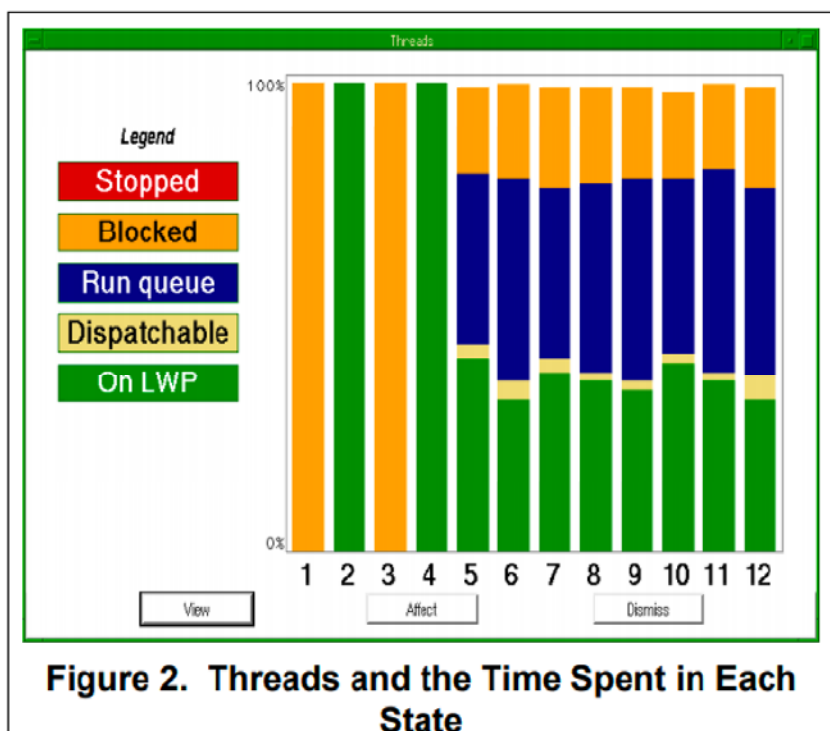
---

<sup>4</sup> WJ Buchanan, M Naylor and AV Scott, Enhancing Network Management using Mobile Agents (published January 10, 1997) (“Buchanan”).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Debugging the scheduler with conventional tools was difficult-it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2).



(Cantrill at FIG. 2)

### Buchanan

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents.* These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development. To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them. (Buchanan at Abstract).

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation. A server application that could only handle a request from one client would be of limited use. Threads provide a means to allow an application to perform multiple tasks simultaneously.*** Java makes creating, controlling, and co-ordinating threads relatively simple. The main advantages of threads are:

- o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data. For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.***

- o They are easier to test, as ***each thread can be tested independently of other threads.***

- o They can use standard threads, which are optimised for given hardware. (Buchanan at sec. 6).

Traditional client/server architectures are typically wasteful in their usage of bandwidth. ***Agent mobility overcomes this by minimizing bandwidth consumption, as they support:***

- ***Adaptive network load balancing.***



HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY

- Solve problems caused by intermittent or unreliable network connections. (Buchanan at sec. 1).

(Buchanan at FIG. 1).

There are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.* (Buchanan at sec. 4).

3. Claim 3

- a. The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.

920. NetRanger discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an

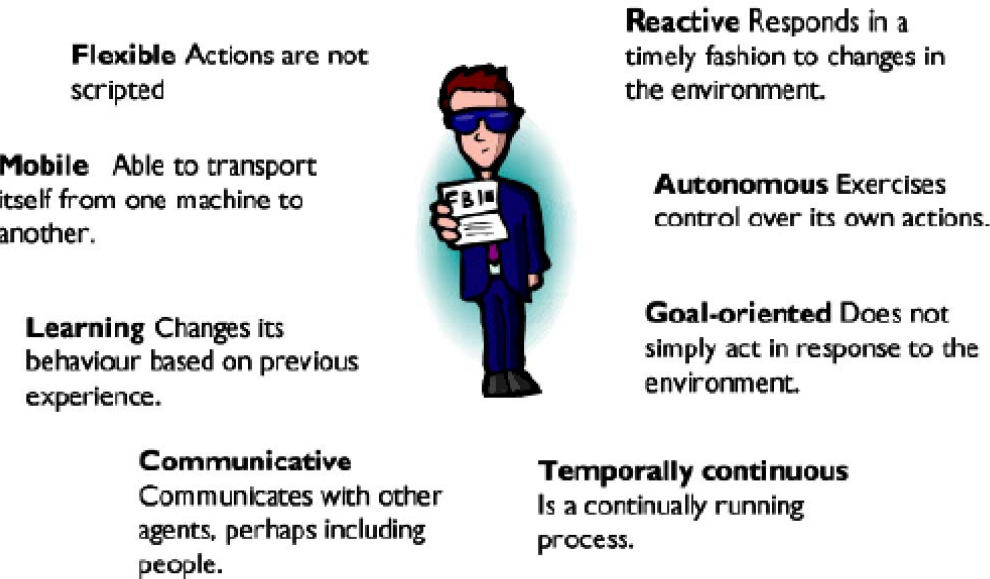


Figure 1 Agent properties

abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

NetRanger is not only an Intrusion Detection system, it also looks for patterns of misuse, or attack signatures. Patterns can be as simple as an attempt to access a specific port on a specific host, or as complex as sequences of operations distributed across multiple hosts over an arbitrary period of time. The first type of pattern is termed an atomic signature; the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

second, a composite signature. Often, NetRanger suspicious activity and alert security personnel long before an attack occurs.

2.1.1 User's Guide at 1-6.

921. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

922. NetRanger discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example:

Large ICMP traffic. Numerous computers are vulnerable to an attack where if you send an ICMP packet with an extremely large data size it will crash the machine. NetRanger blocks and alarms this traffic.

2.1.1 User's Guide at 4-65

Half-Open SYN Attack. This attack was recently publicized when it was used to shut down several Internet Service Providers. This attack can crash a machine by overloading it with TCP connection requests that it never closes.

2.1.1 User's Guide at 4-65; *see also id.* at 4-74, 4-75.

923. It would have been obvious to combine NetRanger with the OSPF routing protocol (RFC 2328). Under NetFuel's interpretation, triggering policies based off OSPF events satisfies this limitation. NetRanger monitors ICMP messages that change the routing table, and the OSPF routing protocol was the prevalent gateway routing protocol of the time. It would have been obvious at the time of invention to one of ordinary skill in the art to modify NetRanger to include attack signatures that use changes to the OSPF routing table as a component to indicate potential malicious activity. I am familiar with the Internet Engineering Task Force ("IETF"), and have been an active user of this service and website throughout my career. The IETF is an Internet standards body, comprised of network designers, operators, vendors, and researchers. The IETF

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

publishes Requests For Comments (“RFC”), which are memoranda describing methods, behaviors, research, or innovations concerning the workings of the Internet. Drafts of the RFCs are published, and publicly available on the IETF website. Information about the Internet Standards Process can be found in RFC 2026: <https://datatracker.ietf.org/doc/rfc2026/>.

924. The IETF Datatracker is the primary day-to-day front-end to the IETF database, and stores documentation about the RFCs. RFC2028 (OSPF Version 2) was publicly available at least as early as the earliest priority date of the ’730 Patent and can be found at the following link: <https://tools.ietf.org/html/rfc2328>. T Based on my knowledge of the IETF, when this RFC was published it would have been publicly available on the IETF website.

925. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

926. NetRanger discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

The NetRanger CD contains software to configure and install a NetRanger Sensor on either a x86 or S PARC Solaris workstation.

Minimum Requirements

Before installing a new NetRanger Sensor, prepare the workstation by installing the Solaris OS. Do not install the NetRanger Sensor software on a workstation that does not meet the minimum requirements outlined in the Minimum Requirements section of the NetRanger OS Installation Instructions.

2.1.1 User’s Guide at 3-8, 3-14. *See supra* claim limitation 1.0.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *running at least one thread in a first runtime environment;*

927. NetRanger discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

928. NetRanger discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

929. NetRanger discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- e. *and performing a corrective action to fix any detected abnormalities;*

930. NetRanger discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

931. NetRanger discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

932. NetRanger discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

933. NetRanger discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

934. NetRanger discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

935. NetRanger discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

NetRanger is an intrusion detection system that can be plugged into your TCP/IP networks in a variety of ways. It detects and responds to unauthorized activity in real time. Unauthorized activity includes attempts to circumvent an existing firewall, router, and network security policies, as well as misuse of authorized services. NetRanger is an enterprise-wide solution that can monitor a large number of networks via centralized management. As shown in Figure 1.1, NetRanger consists of three basic components: a Sensor, a communication system, and a Director.

2.1.1 User’s Guide at 1-1; *see also id.* at 1-2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

936. NetRanger discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13.

For example:

The Director

The Director displays events, monitors Sensors, and analyzes data. It communicates with one or more Sensor systems via the Communication System. The Director contains two basic subsystems: the Security Management I nterface (SMI) and the Data Management Package (DMP).

2.1.1 User’s Guide at 1-5; *see also id.* at 1-1,1-2.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

937. NetRanger discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

The Director

The Director displays events, monitors Sensors, and analyzes data. It communicates with one or more Sensor systems via the Communication System. The Director contains two basic subsystems: the Security Management I nterface (SMI) and the Data Management Package (DMP).

2.1.1 User’s Guide at 1-5; *see also id.* at 1-1,1-2.

d. *running at least one thread in a first runtime environment;*

938. NetRanger discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

939. NetRanger discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

940. NetRanger discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

941. NetRanger discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

942. NetRanger discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

943. NetRanger discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

944. NetRanger discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

945. NetRanger discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

946. NetRanger discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

947. NetRanger discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

948. I expect to testify that this art anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

**S. Anticipation by and/or Obviousness in View of the Cisco Catalyst 5000**



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**Switch Family, marketed by Cisco by June 1995 at the latest.**

1. ***Claim 1***

a. *A computer-implemented method, comprising:*

949. Cisco Catalyst 5000 Switch Family discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

**Unique Traffic Management Capabilities**

Embedded traffic management functionality is a critical component of the Catalyst 5000 design. The Catalyst 5000 supports extensive input buffers across its switching modules that function with buffer management software to eliminate packet loss during peak traffic periods.

The Catalyst 5000 architecture supports multiple levels of data prioritization, with each interface separately user-configurable as high or low priority. Maintenance of separate logical queues for each priority class reduces buffering delays, a critical issue with typically bursty network traffic and delay-sensitive traffic types such as voice and video.

The Catalyst 5000 family supports switched port analyzer software. This software gives network managers the ability to track traffic on any one of the 5000's switching ports.

Cisco is the only vendor to support embedded remote monitoring (RMON) across all of its switching platforms. Because RMON is fully integrated with the CiscoView device management application, network managers can view, configure and set alarms on a graphical representation of the switch.

Catalyst 5000 also supports RFC 1271 standard-compliant RMON across all ports, enabling users to centrally manage, administer and control traffic monitoring for all switched LAN segments simultaneously. Offering centralized control that greatly reduces administrative overhead, the embedded RMON agent can save the user up to 80 percent of the cost of using specialized hardware probes on each LAN segment.

"Cisco Delivers Switching for Wiring Closets" March 28, 1995.

b. *running at least one thread in a first runtime environment;*

950. Cisco Catalyst 5000 Switch Family discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example, Cisco Catalyst 5000 supports multithreaded execution. To the extent this limitation is not met by Cisco Catalyst 5000 switch family, it would have been obvious at the time of invention to run multiple threads in IOS. *See, e.g.*, Exs. B-1, B-12 at limitation 1.1.

c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

951. Cisco Catalyst 5000 Switch Family discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example, Cisco Catalyst 5000 supports multithreaded execution and monitoring CPU usage for each of those threads. To the extent this limitation is not expressly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time for managed networks to monitor per-thread utilization in order to, for example, perform load-balancing operations.

d. *detecting if there is an abnormality in the monitored operational parameters;*

952. Cisco Catalyst 5000 Switch Family discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and Berry or Cantrill. The motivation of such an modification would be to provide a system that monitors CPU utilization by a particular thread (as taught by Berry and Cantrill) to determine whether it matches an attack signature for detecting intrusion, as taught by NetRanger. *See* Ex. B-8, limitation 1.3.

e. *and performing a corrective action to fix any detected abnormalities,*

953. Cisco Catalyst 5000 Switch Family discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and Berry or Cantrill. The motivation of such a modification would be to provide a system that monitors CPU utilization by a particular thread (as taught by Berry and Cantrill) to determine whether it matches an attack signature for detecting intrusion, as taught by NetRanger. *See* Ex. B-8, limitation 1.4.

wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

954. Cisco Catalyst 5000 Switch Family discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and Berry or Cantrill. The motivation of such am modification would be to provide a system that monitors CPU utilization by a particular thread (as taught by Berry and Cantrill) to determine whether it matches an attack signature for detecting intrusion, as taught by NetRanger. *See* Ex. B-8, limitation 1.5.

- f. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

955. Cisco Catalyst 5000 Switch Family discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and Berry or Cantrill. The motivation of such am modification would be to provide a system that monitors CPU utilization by a particular thread (as taught by Berry and Cantrill) to determine whether it matches an attack signature for detecting intrusion, as taught by NetRanger. *See* Ex. B-8, limitation 1.6.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

956. Cisco Catalyst 5000 Switch Family discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and Berry or Cantrill. The motivation of such a modification would be to provide a system that monitors CPU utilization by a particular thread (as taught by Berry and Cantrill) to determine whether it matches an attack signature for detecting intrusion, as taught by NetRanger. *See* Ex. B-8, limitation 1.7.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

957. Cisco Catalyst 5000 Switch Family discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and Berry or Cantrill. The motivation of such a modification would be to provide a system that monitors CPU utilization by a particular thread (as taught by Berry and Cantrill) to determine whether it matches an attack signature for detecting intrusion, as taught by NetRanger. *See* Ex. B-8, limitation 1.3.

Claim 3

- b. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

958. Cisco Catalyst 5000 Switch Family discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and Berry ’654 or Cantrill. The motivation of such a modification would be to provide a system that monitors CPU utilization by a particular thread (as taught by Berry ’654 and Cantrill) to determine whether it matches an attack signature for detecting intrusion, as taught by NetRanger. *See* Ex. B-8, limitation 3.0.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

3. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

959. Cisco Catalyst 5000 Switch Family discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example, to the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger and OSPF (RFC 2328). *See* Ex. B-8, limitation 6.0.

Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

960. Cisco Catalyst 5000 Switch Family discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. For example:

**Unique Traffic Management Capabilities**

Embedded traffic management functionality is a critical component of the Catalyst 5000 design. The Catalyst 5000 supports extensive input buffers across its switching modules that function with buffer management software to eliminate packet loss during peak traffic periods.

The Catalyst 5000 architecture supports multiple levels of data prioritization, with each interface separately user-configurable as high or low priority. Maintenance of separate logical queues for each priority class reduces buffering delays, a critical issue with typically bursty network traffic and delay-sensitive traffic types such as voice and video.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The Catalyst 5000 family supports switched port analyzer software. This software gives network managers the ability to track traffic on any one of the 5000's switching ports.

Cisco is the only vendor to support embedded remote monitoring (RMON) across all of its switching platforms. Because RMON is fully integrated with the CiscoView device management application, network managers can view, configure and set alarms on a graphical representation of the switch.

Catalyst 5000 also supports RFC 1271 standard-compliant RMON across all ports, enabling users to centrally manage, administer and control traffic monitoring for all switched LAN segments simultaneously. Offering centralized control that greatly reduces administrative overhead, the embedded RMON agent can save the user up to 80 percent of the cost of using specialized hardware probes on each LAN segment.

"Cisco Delivers Switching for Wiring Closets" March 28, 1995.

b. *running at least one thread in a first runtime environment;*

961. Cisco Catalyst 5000 Switch Family discloses this element of claim 7, "running at least one thread in a first runtime environment," at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

962. Cisco Catalyst 5000 Switch Family discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

963. Cisco Catalyst 5000 Switch Family discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

964. Cisco Catalyst 5000 Switch Family discloses this element of claim 7, "performing a corrective action to fix any detected abnormalities," at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

965. Cisco Catalyst 5000 Switch Family discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

966. Cisco Catalyst 5000 Switch Family discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**5. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

967. Cisco Catalyst 5000 Switch Family discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

**6. Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

968. Cisco Catalyst 5000 Switch Family discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

7. ***Claim 13***

a. *A system, comprising:*

969. Cisco Catalyst 5000 Switch Family discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

Unique Traffic Management Capabilities

Embedded traffic management functionality is a critical component of the Catalyst 5000 design. The Catalyst 5000 supports extensive input buffers across its switching modules that function with buffer management software to eliminate packet loss during peak traffic periods.

The Catalyst 5000 architecture supports multiple levels of data prioritization, with each interface separately user-configurable as high or low priority. Maintenance of separate logical queues for each priority class reduces buffering delays, a critical issue with typically bursty network traffic and delay-sensitive traffic types such as voice and video.

The Catalyst 5000 family supports switched port analyzer software. This software gives network managers the ability to track traffic on any one of the 5000's switching ports.

Cisco is the only vendor to support embedded remote monitoring (RMON) across all of its switching platforms. Because RMON is fully integrated with the CiscoView device management application, network managers can view, configure and set alarms on a graphical representation of the switch.

Catalyst 5000 also supports RFC 1271 standard-compliant RMON across all ports, enabling users to centrally manage, administer and control traffic monitoring for all switched LAN segments simultaneously. Offering centralized control that greatly reduces administrative overhead, the embedded RMON agent can save the user up to 80 percent of the cost of using specialized hardware probes on each LAN segment.

"Cisco Delivers Switching for Wiring Closets" March 28, 1995.

b. *a processor;*

970. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

Unique Traffic Management Capabilities

Embedded traffic management functionality is a critical component of the Catalyst 5000 design. The Catalyst 5000 supports extensive input buffers



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

across its switching modules that function with buffer management software to eliminate packet loss during peak traffic periods.

The Catalyst 5000 architecture supports multiple levels of data prioritization, with each interface separately user-configurable as high or low priority. Maintenance of separate logical queues for each priority class reduces buffering delays, a critical issue with typically bursty network traffic and delay-sensitive traffic types such as voice and video.

The Catalyst 5000 family supports switched port analyzer software. This software gives network managers the ability to track traffic on any one of the 5000's switching ports.

Cisco is the only vendor to support embedded remote monitoring (RMON) across all of its switching platforms. Because RMON is fully integrated with the CiscoView device management application, network managers can view, configure and set alarms on a graphical representation of the switch.

Catalyst 5000 also supports RFC 1271 standard-compliant RMON across all ports, enabling users to centrally manage, administer and control traffic monitoring for all switched LAN segments simultaneously. Offering centralized control that greatly reduces administrative overhead, the embedded RMON agent can save the user up to 80 percent of the cost of using specialized hardware probes on each LAN segment.

"Cisco Delivers Switching for Wiring Closets" March 28, 1995.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

971. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, "and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising." '659 Pat. at Cl. 13. For example:

#### Unique Traffic Management Capabilities

Embedded traffic management functionality is a critical component of the Catalyst 5000 design. The Catalyst 5000 supports extensive input buffers across its switching modules that function with buffer management software to eliminate packet loss during peak traffic periods.

The Catalyst 5000 architecture supports multiple levels of data prioritization, with each interface separately user-configurable as high or low priority. Maintenance of separate logical queues for each priority class reduces buffering delays, a critical issue with typically bursty network traffic and delay-sensitive traffic types such as voice and video.

The Catalyst 5000 family supports switched port analyzer software. This software gives network managers the ability to track traffic on any one of the 5000's switching ports.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cisco is the only vendor to support embedded remote monitoring (RMON) across all of its switching platforms. Because RMON is fully integrated with the CiscoView device management application, network managers can view, configure and set alarms on a graphical representation of the switch.

Catalyst 5000 also supports RFC 1271 standard-compliant RMON across all ports, enabling users to centrally manage, administer and control traffic monitoring for all switched LAN segments simultaneously. Offering centralized control that greatly reduces administrative overhead, the embedded RMON agent can save the user up to 80 percent of the cost of using specialized hardware probes on each LAN segment.

"Cisco Delivers Switching for Wiring Closets" March 28, 1995.

d. *running at least one thread in a first runtime environment;*

972. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, "running at least one thread in a first runtime environment," at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

973. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, "monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread," at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

974. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, "detecting there is an abnormality in the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

975. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, "and performing a corrective action to fix any detected abnormalities," at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

976. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

977. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

978. Cisco Catalyst 5000 Switch Family discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

8. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

979. Cisco Catalyst 5000 Switch Family discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

9. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises comparing the monitored operational parameters to known thresholds.*

980. Cisco Catalyst 5000 Switch Family discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

10. ***Claim 18***

a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

981. Cisco Catalyst 5000 Switch Family discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

982. I expect to testify that this art anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

**T.**

[REDACTED]

1. ***Claim 1***

a. *A computer-implemented method, comprising:*

983. [REDACTED] discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

[REDACTED]

[REDACTED]

- [REDACTED]

[REDACTED]

[REDACTED]

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

[REDACTED]

[REDACTED]

CSI-NF-00000014 at 2; *see also* CSI-NF-00000005 at 6, 8.

984. Further examples are also found at

b. *running at least one thread in a first runtime environment;*

[REDACTED] discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

[REDACTED]

CSI-NF-00000006 at 8; *see id* at CSI-NF-00000005 at 7, 11.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

985. [REDACTED] this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example, to the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of \_\_\_\_.

d. *detecting if there is an abnormality in the monitored operational parameters;*

986. [REDACTED] discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

example, to the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of \_\_\_\_.

- e. *and performing a corrective action to fix any detected abnormalities,*

987. [REDACTED] discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example, to the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of \_\_\_\_.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

988. [REDACTED] discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example, to the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of \_\_\_\_.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

989. [REDACTED] discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example, to the extent [REDACTED] [REDACTED] not expressly or implicitly disclose this limitation, it would have been obvious in view of \_\_\_\_.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

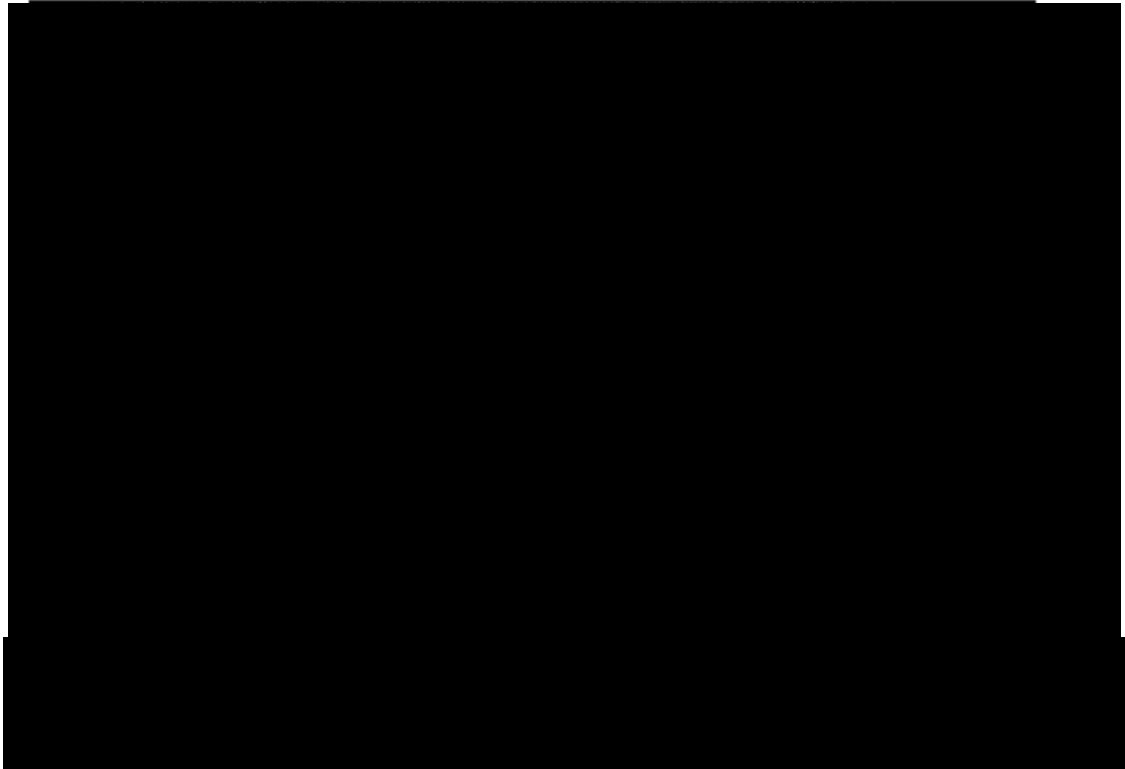
*multiple threads.*

990. [REDACTED] discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example, to the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of \_\_\_\_.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

991. [REDACTED] discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:



CSI-NF-00000005 at 7; *see also* CSI-NF-00000014 at 3; CSI-NF-00000006 at 6.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

992. [REDACTED] discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3 at least for all the reasons explained above regarding claim 1, element c.

993. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

994. [REDACTED] discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6.

995. Furthermore, it would have been obvious to combine [REDACTED]  
[REDACTED] Under NetFuel’s interpretation, OSPF packets as triggering events and initiating a policy in response satisfies this limitation. [REDACTED]  
[REDACTED]

[REDACTED]. OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement [REDACTED]

[REDACTED] also CSI-NF-00000014 at 2.

996. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

5. *Claim 7*

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

997. [REDACTED] discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED] 00000014 at 2; *see also* CSI-NF-00000005 at 6, 8.

- b. *running at least one thread in a first runtime environment;*

998. [REDACTED] discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

999. [REDACTED] discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

1000. [REDACTED] discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- e. *and performing a corrective action to fix any detected abnormalities;*

1001. [REDACTED] discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1002. [REDACTED] discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1003. [REDACTED] discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. **Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

1004. [REDACTED] discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1005. [REDACTED] discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

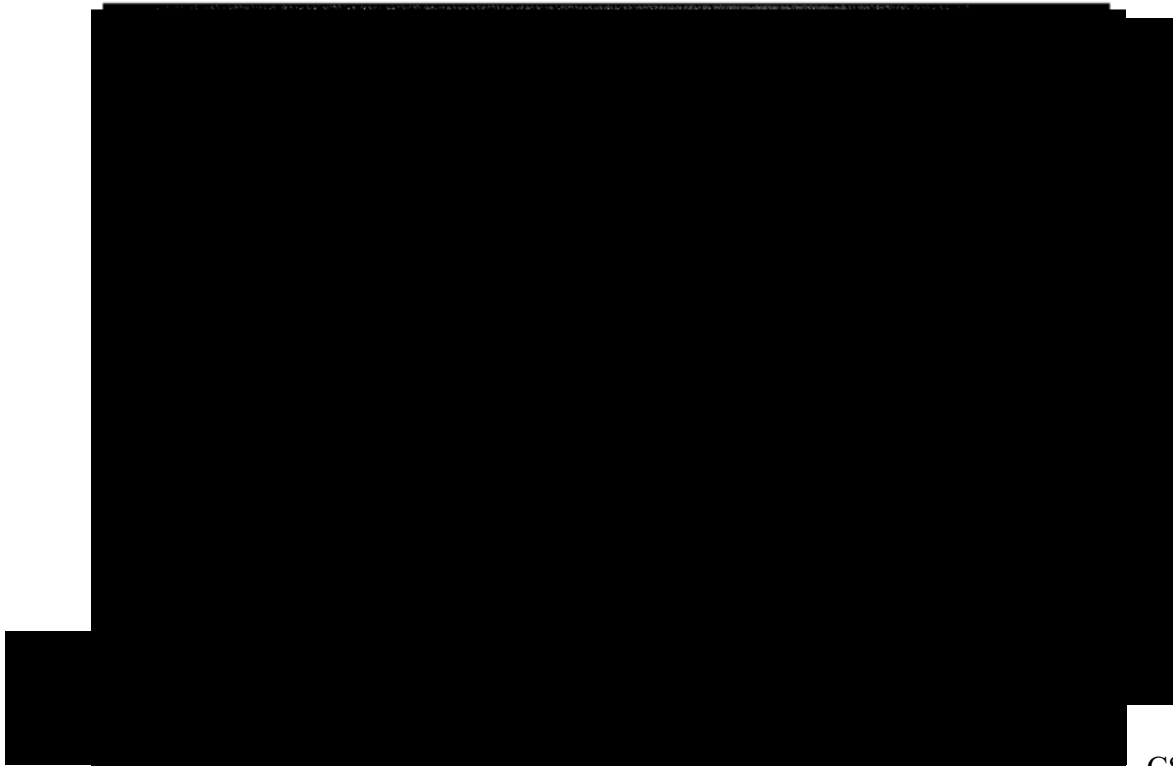
- a. *A system, comprising:*

1006. [REDACTED] otection discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

- b. *a processor;*

1007. [REDACTED] discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-

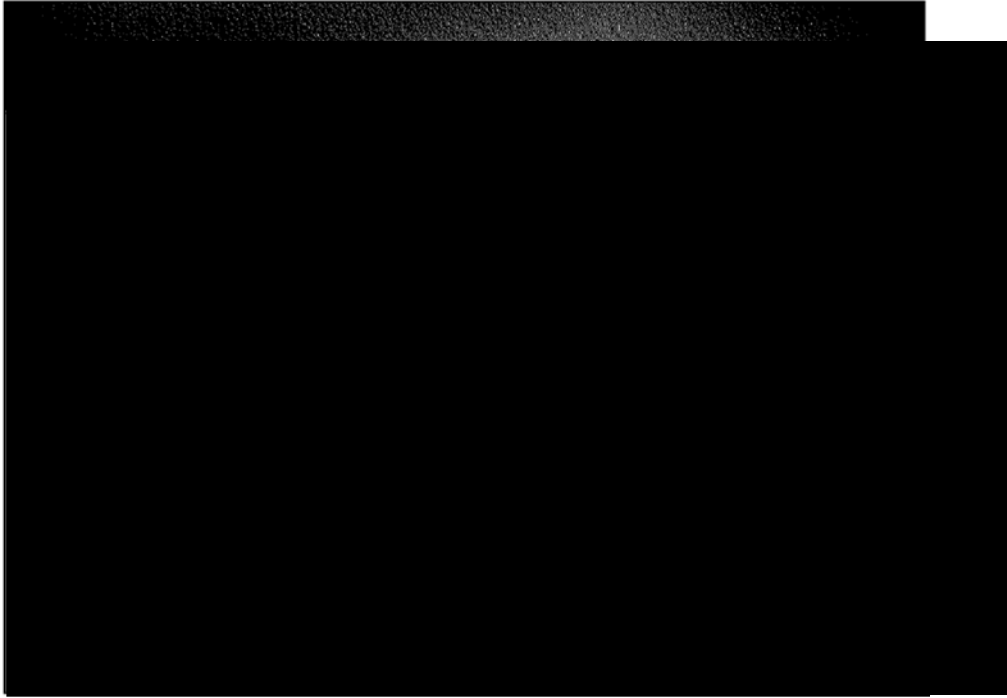
00000005 at 7

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1008. [REDACTED] discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

CSI-NF-00000005 at 6; *see also id.* at 8.



d. *running at least one thread in a first runtime environment;*

1009. [REDACTED] discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1010. [REDACTED] discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

1011. [REDACTED] discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

1012. [REDACTED] discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1013. [REDACTED] discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1014. [REDACTED] discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1015. [REDACTED] discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1016. [REDACTED] discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1017. [REDACTED] discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1018. [REDACTED] discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

I expect to testify that [REDACTED] each Asserted Claim of the ’659 Patent. A claim chart illustrating that [REDACTED] discloses each and every limitation of those claims is included as Ex. B-10.

**U. Anticipation by and/or Obviousness in View of Cisco Receive ACL (“rACL”), marketed by Cisco by April 22, 2002 at the latest.**

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

1019. Cisco rACL discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

[REDACTED]

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



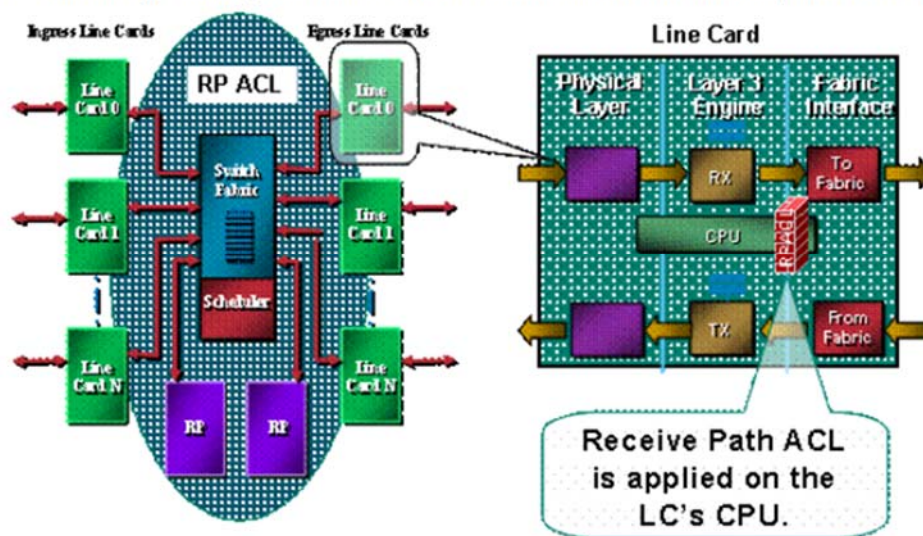
CSI-NF-00000013 at 4; *see also id.* at 12; CSI-NF-00000018 at 2; CSI-NF-00000025.

1020. Further examples are also found at

b. *running at least one thread in a first runtime environment;*

1021. Cisco rACL discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

As this image shows, the rACL is executed on each LC before the packet is transmitted to the GRP.





**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

GSR: Receive Access Control Lists<sup>5</sup> at 2-3.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1022. Cisco rACL discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

1. Identify protocols used in the network with a classification ACL.

Deploy an rACL that permits all the known protocols that access the GRP. This “discovery” rACL should have both source and destination addresses set to any. Logging can be used to develop a list of source addresses that match the protocol permit statements. In addition to the protocol permit statement, a permit any any log line at the end of the rACL can be used to identify other protocols that would be filtered by the rACL and that might require access to the GRP.

The objective is to determine what protocols the specific network uses. Logging should be used for analysis to determine “what else” might be communicating with the router.

Note: Although the log keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses this keyword might result in an overwhelming number of log entries and possibly high router CPU usage. Use the log keyword for short periods of time and only when needed to help classify traffic.

2. Review identified packets and begin to filter access to the GRP.

Once the packets filtered by the rACL in step 1 have been identified and reviewed, deploy an rACL with a permit any any statement for the allowed protocols. Just as in step 1, the log keyword can provide more information about the packets that match the permit entries. Using deny any any log at the end can help identify any unexpected packets destined to the GRP. This rACL will provide basic protection and will allow network engineers to ensure that all required traffic is permitted.

The objective is to test the range of protocols that need to communicate with the router without having the explicit range of IP source and destination addresses.

GSR: Receive Access Control Lists at 7–8.

---

<sup>5</sup> GSR: Receive Access Control Lists, Cisco Systems, Inc. (Sept. 12, 2018) <https://www.cisco.com/c/en/us/support/docs/ip/access-lists/43861-racl.html> (last updated Feb. 23, 2006).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1023. To the extent this limitation is not disclosed in Cisco Receive ACL, it is rendered obvious in view of at least NetRanger (see Ex. B-8, limitation 1.2), Cantrill and/or U.S. Patent No. 6,658,654 (“Berry ’654”). NetRanger discloses agents with one or more associated underlying resources and monitoring events and parameters associated with the underlying resources. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry ’654. For example:

Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used. In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests. Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry ’654 at 1:18–36.

Performance monitoring is often used to optimize the use of software in a system.

Berry ’654 at 2:5–6.

A method, system, apparatus, or computer program product is presented for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment. While a first set of events is being gathered or monitored for a particular thread as a first metric, events that may indirectly cause inaccuracies in the first metric are also monitored as a second metric, and the presence of a positive value for the second metric is then used to determine that the first metric is inaccurate or unreliable. If the first metric is deemed inaccurate, then the first metric may be discarded. When the first metric is gathered without the occurrence of an event being counted as the second metric, then the first metric is considered accurate. The first metric may then be considered a “thread-relative” metric as it has been observed during a time period in which no events would have caused the first metric to become inaccurate during the execution of a particular thread. For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

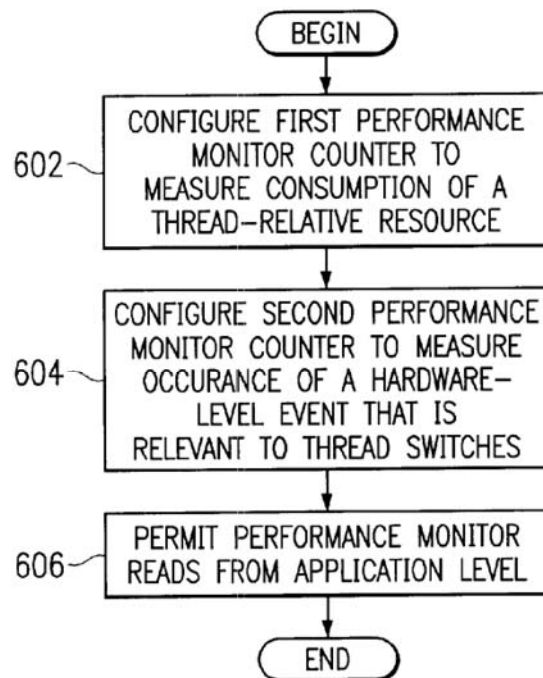
is a number of interrupts, each of which might cause the kernel to initiate a thread switch.

Berry '654 at 3:42–64.

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry '654 at 12:65–13:10.

Berry '654 at Fig. 6A.



*FIG. 6A*

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance—thus its minimization is an important goal. By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources. Compounding this resource-

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.

Cantrill at 253.

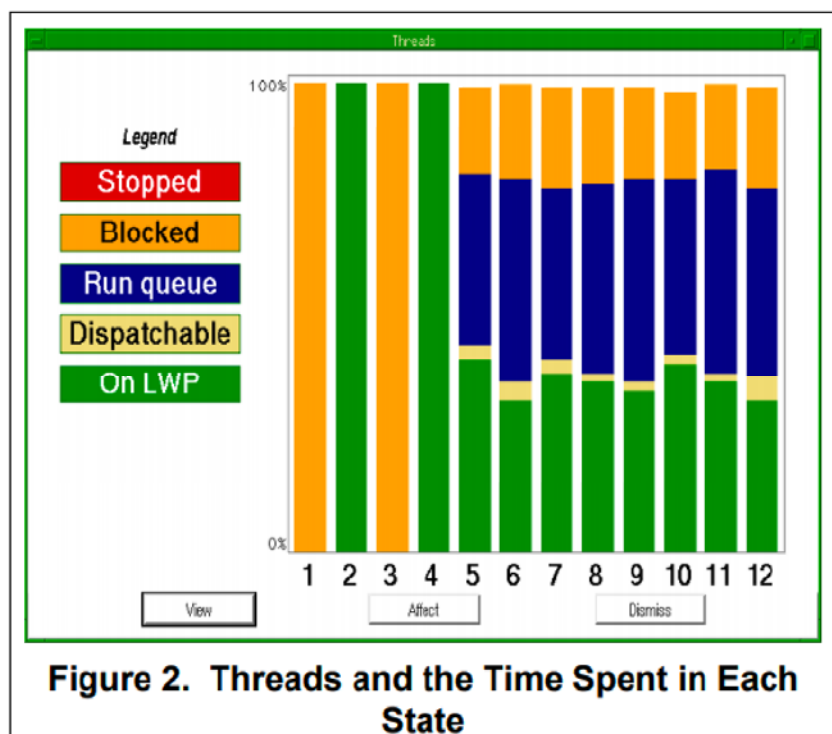
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at 260.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2.



d. *detecting if there is an abnormality in the monitored operational parameters;*

1024. Cisco rACL discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

1. Identify protocols used in the network with a classification ACL.

Deploy an rACL that permits all the known protocols that access the GRP. This “discovery” rACL should have both source and destination addresses set to any. Logging can be used to develop a list of source addresses that match the protocol permit statements. In addition to the protocol permit statement, a permit any any log line at the end of the rACL can be used to identify other protocols that would be filtered by the rACL and that might require access to the GRP.

The objective is to determine what protocols the specific network uses. Logging should be used for analysis to determine “what else” might be communicating with the router.

Note: Although the log keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses this keyword might result in an overwhelming number of log entries and possibly high router CPU usage. Use the log keyword for short periods of time and only when needed to help classify traffic.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2. Review identified packets and begin to filter access to the GRP.

Once the packets filtered by the rACL in step 1 have been identified and reviewed, deploy an rACL with a permit any any statement for the allowed protocols. Just as in step 1, the log keyword can provide more information about the packets that match the permit entries. Using deny any any log at the end can help identify any unexpected packets destined to the GRP. This rACL will provide basic protection and will allow network engineers to ensure that all required traffic is permitted.

The objective is to test the range of protocols that need to communicate with the router without having the explicit range of IP source and destination addresses.

GSR: Receive Access Control Lists at 7–8.

1025. To the extent this limitation is not disclosed by Cisco rACL, it is rendered obvious by at least NetRanger. See Ex. B-8 at limitation 1.3.

e. *and performing a corrective action to fix any detected abnormalities,*

1026. Cisco rACL discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

[REDACTED]

CSI-NF-00000020.

Traffic that enters an LC is first sent to the local CPU of the LC, and packets that require processing by the GRP are queued for forwarding to the route processor. The receive ACL is created on the GRP and then pushed down to the CPUs of the various LCs. Before traffic is sent from the LC CPU to the GRP, the traffic is compared to the rACL. If permitted, the traffic passes to the GRP, while all other traffic is denied. The rACL is inspected prior to the LC to GRP rate-limiting function. Since the rACL is used for all receive adjacencies, some packets that are handled by the LC CPU (such as echo requests) are subject to rACL filtering as well. This needs to be taken into account when designing rACL entries.



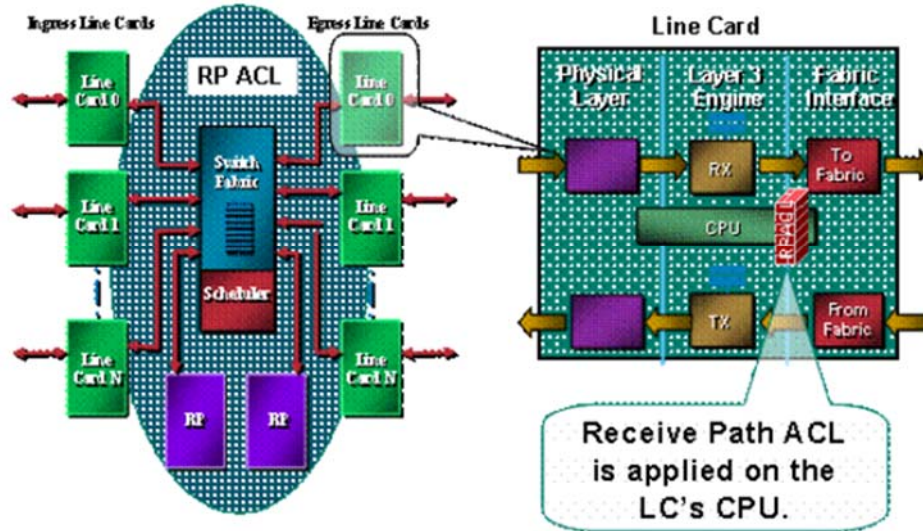
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

GSR: Receive Access Control Lists at 3; *see also* CSI-NF-00000023.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1027. Cisco rACL discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1.

GSR: Receive Access Control Lists at 2, *see also* CSI-NF-00000032 at 2; CSI-NF-  
As this image shows, the rACL is executed on each LC before the packet is transmitted to the GRP.



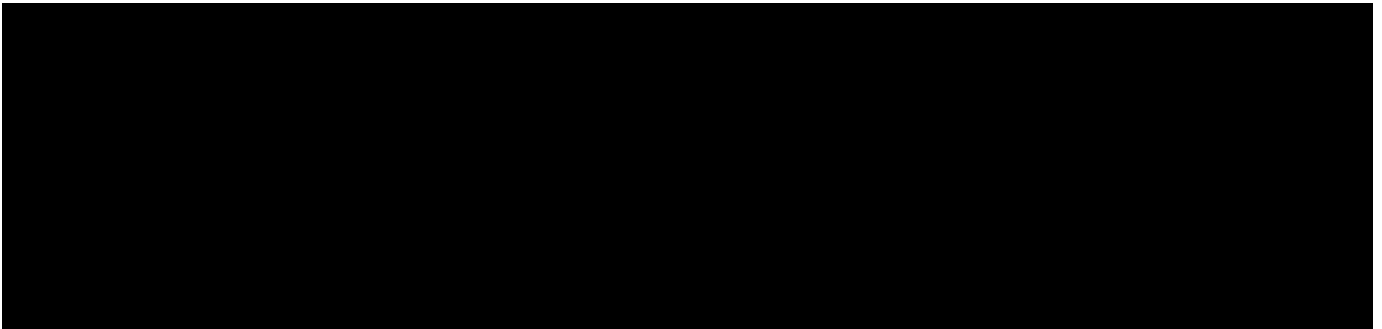
00000033 at 2.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1028. Cisco rACL discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

CSI-NF-00000032 at 2.



- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

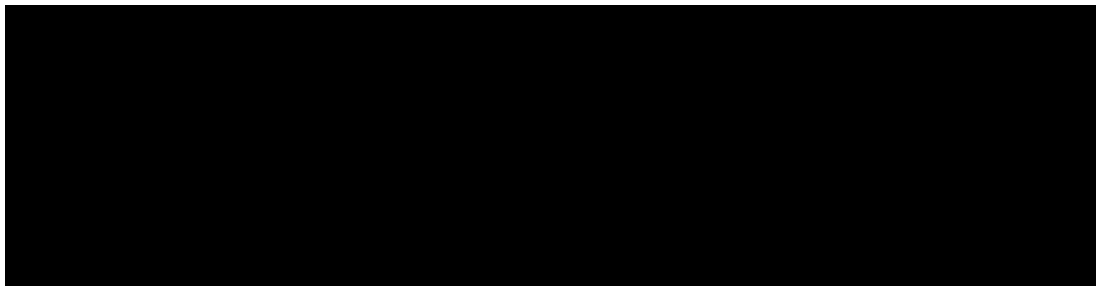
1029. Cisco rACL discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

1030. To the extent this limitation is not disclosed by Cisco rACL, it is rendered obvious by at least NetRanger. *See* Ex. B-8 at limitation 1.7.

**2. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

1031. Cisco rACL discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:



CSI-NF-00000013 at 12.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1032. Furthermore, as discussed above, this limitation is rendered obvious by at least Cantrill.

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance—thus its minimization is an important goal. By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources. Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.

Cantrill at 253.

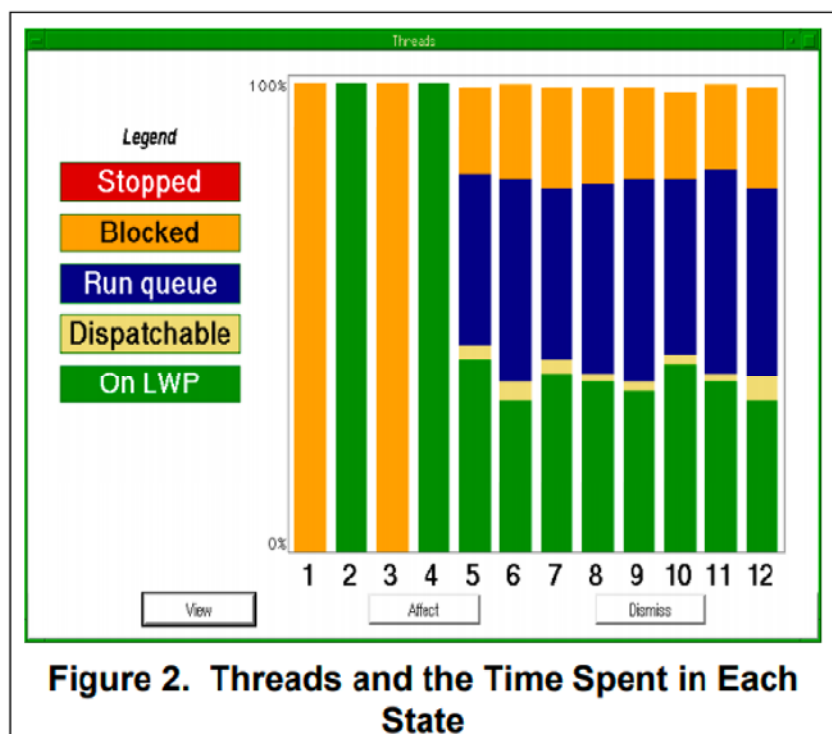
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at 260.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2.



3. **Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1033. Cisco rACL discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

1. Identify protocols used in the network with a classification ACL.

Deploy an rACL that permits all the known protocols that access the GRP. This “discovery” rACL should have both source and destination addresses set to any. Logging can be used to develop a list of source addresses that match the protocol permit statements. In addition to the protocol permit statement, a permit any any log line at the end of the rACL can be used to identify other protocols that would be filtered by the rACL and that might require access to the GRP.

The objective is to determine what protocols the specific network uses. Logging should be used for analysis to determine “what else” might be communicating with the router.

Note: Although the log keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses this keyword might

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

result in an overwhelming number of log entries and possibly high router CPU usage. Use the log keyword for short periods of time and only when needed to help classify traffic.

2. Review identified packets and begin to filter access to the GRP.

Once the packets filtered by the rACL in step 1 have been identified and reviewed, deploy an rACL with a permit any any statement for the allowed protocols. Just as in step 1, the log keyword can provide more information about the packets that match the permit entries. Using deny any any log at the end can help identify any unexpected packets destined to the GRP. This rACL will provide basic protection and will allow network engineers to ensure that all required traffic is permitted.

The objective is to test the range of protocols that need to communicate with the router without having the explicit range of IP source and destination addresses.

GSR: Receive Access Control Lists at 7–8; *see also* CSI-NF-00000023 at 2.

1034. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

1035. Cisco rACL discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example:

Receive Adjacencies and Punted Packets

As described earlier in this document, some packets require GRP processing. The packets are punted from the data-forwarding plane to the GRP. This is a list of the common forms of Layer 3 data that require GRP access.

- Routing protocols
- Multicast control traffic (OSPF, Hot Standby Router Protocol [HSRP], Tag Distribution Protocol [TDP], Protocol Independent Multicast [PIM], and such)
- Multiprotocol Label Switching (MPLS) packets needing fragmentation

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Packets with certain IP options such as router alert
- First packet of multicast streams
- Fragmented ICMP packets that require reassembly
- All traffic destined to the router itself (except for the traffic handled on the LC)

Since rACLs apply to receive adjacencies, the rACL filters some traffic that is not punted to the GRP but is a receive adjacency. The most common example of this is an ICMP echo request (ping). ICMP echo requests directed to the router are handled by the LC CPU; since the requests are receive adjacencies, they are also filtered by the rACL. Therefore, to allow pings to the interfaces (or loopbacks) of the router, the rACL must explicitly permit the echo requests.

Receive adjacencies can be viewed using the show ip cef command.

GSR: Receive Access Control Lists at 6–7

1036. rACLs are triggered based in part on receive adjacencies such as OSPF traffic, which satisfies this limitation under NetFuel's apparent interpretation of this claim.

1037. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

1038. Cisco rACL discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-00000013 at 4; *see also id.* at 12; CSI-NF-00000018 at 2; CSI-NF-00000025.

b. *running at least one thread in a first runtime environment;*

1039. Cisco rACL discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1040. Cisco rACL discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

1041. Cisco rACL discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *and performing a corrective action to fix any detected abnormalities;*

1042. Cisco rACL discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1043. Cisco rACL discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1044. Cisco rACL discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

1045. Cisco rACL discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1046. Cisco rACL discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

1047. Cisco rACL discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

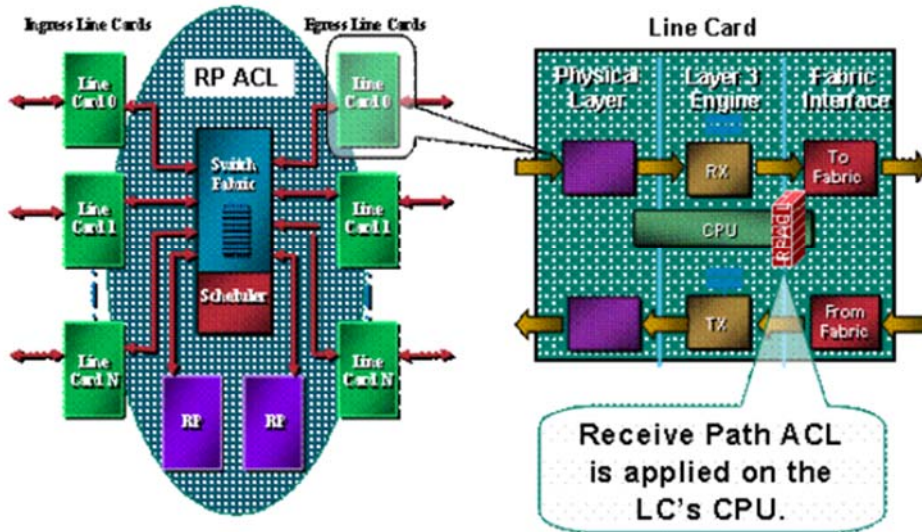


CSI-NF-00000013 at 4; *see also id.* at 12; CSI-NF-00000018 at 2; CSI-NF-00000025.

b. *a processor;*

1048. Cisco rACL discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13.

As this image shows, the rACL is executed on each LC before the packet is transmitted to the GRP.



GSR: Receive Access Control Lists at 2-3.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*system to perform a method comprising:*

1049. Cisco rACL discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:



CSI-NF-00000013 at 4; *see also id.* at 12; CSI-NF-00000018 at 2; CSI-NF-00000025.

d. *running at least one thread in a first runtime environment;*

1050. Cisco rACL discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1051. Cisco rACL discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

1052. Cisco rACL discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

1053. Cisco rACL discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1054. Cisco rACL discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1055. Cisco rACL discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1056. Cisco rACL discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1057. Cisco rACL discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1058. Cisco rACL discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1059. Cisco rACL discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

1060. I expect to testify that Cisco rACL anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Cisco rACL discloses each and every limitation of those claims is included as Ex. B-11.

**V. Anticipation by and/or Obviousness in View of the IBM xSeries Server**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**Software Rejuvenation Agent, marketed by IBM by 2000 at the latest.**

1. ***Claim 1***

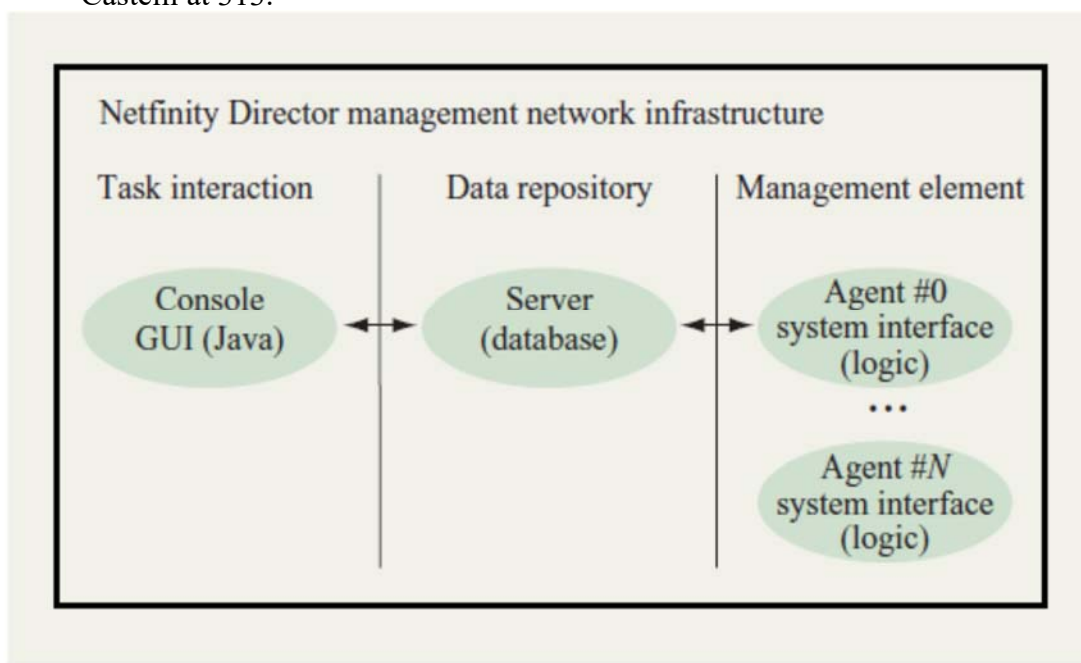
a. *A computer-implemented method, comprising:*

1061. IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim

1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

Castelli at 313.



**Figure 1**

**IBM Director framework.**

Castelli at 315; *see id.* at 312.

b. *running at least one thread in a first runtime environment;*

1062. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim

1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- *Management console*: The IBM Director management console is the graphical user interface (GUI) from which administrative tasks are performed. It is the primary interface with the administrator, and is used to configure the SRA as described below. ***The management console GUI is Java-based, with all state information stored on the server. It runs as a locally installed Java application in a Java Virtual Machine (JVM\*\*).***
- *Management server*: The management server is the platform used for the central management server, where management databases, the server engine, and ***management application logic reside***.
- *IBM Director agents*: The agents ***reside on each managed system*** (such as an xSeries server) and act as passive, ***nonintrusive native applications***. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

Castelli at 315; *see id.* at 317.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1063. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

- *IBM Director agents*: The agents ***reside on each managed system (such as an xSeries server)*** and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

***In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes.*** These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:

- *Inventory*: IBM Director discovers new managed systems, collects the appropriate information about these systems, and stores it in the inventory database. It can then be viewed through either a default or a customized view.
- *Resource monitors*: Resource monitors (**Figure 2**) enable the user to view statistics and usage of critical resources on the network. Information can be collected and monitored on attributes such as CPU, disk, memory, and network. SRA is a specialized instance of a resource monitor.
- *Event management*: Event management (**Figure 3**) enables the user to view a log of events that have occurred for a managed system or group of systems and to create event action plans to associate an event with a desired action, such as sending an e-mail, starting a program, logging to a file, or invoking rejuvenation. When the SRA has detected an

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

impending resource exhaustion, it generates events that can be viewed using this functionality

Castelli at 315-316; *see id.* at 312, 14, 17.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

1064. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl.

1. For example:

***Prediction algorithms***

In the current version of SRA, rejuvenation can be based on elapsed time since the last rejuvenation, or on prediction of impending exhaustion.

When using timed rejuvenation, the user interface of IBM Director is used to schedule and perform rejuvenation at a period specified by the user. A calendar interface allows the user to select when to rejuvenate different nodes of the cluster, and to select “blackout” times during which no rejuvenation is to be allowed. Although this sounds rather primitive, our analysis (presented below) shows that for typical clusters that undergo aging, system availability can be improved significantly via this technique.

Castelli at 317; *see id.* at 317-18, 23.

- e. *and performing a corrective action to fix any detected abnormalities,*

1065. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

**Rejuvenation granularity**

When an exhaustion has been predicted, the question arises as to what portion of the environment should be rejuvenated. ***The simple approach to rejuvenation is to perform a node reboot.*** A single computer node is a distinct and compartmentalized unit of the user’s resource environment, and the cluster-management framework within which we are operating should handle such node failures quite adequately. The drawback is that an entire node rejuvenation may be considered too broad in scope, and the time to reboot may be significant. Perhaps only one application on that system, only one of that application’s services, or perhaps even a particular subprocess of that service is the cause of the pending exhaustion. The narrower the scope of rejuvenation, the smaller the disruption will be to the user’s business objective. ***From a functional point of view, the operating system provides various APIs to perform rejuvenation at all of these various levels.*** Whether the resource to be rejuvenated is at the system level, the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

application level, or even the service level, it is likely that the planned downtime (i.e., the rejuvenation) is always less disruptive than an unplanned node outage.

Because of these considerations, the Windows SRA offers two levels of rejuvenation, depending on the scope of the resource exhaustion (i.e., whether an operating system- level exhaustion or an application-level exhaustion has been identified). ***The desired level can be selected using an advanced submenu of the user interface described above.***

- Level 1 is a ***service-level rejuvenation***. Generally, it can be assumed that applications written as a service are written such that a stoppage of that service will save any necessary data (both user and application) and the corresponding restart of that service will bring the application to a state of usability. In such cases, a graceful rejuvenation of that service can be performed.
- Level 2 is an ***operating-system-level rejuvenation (i.e., a reboot)***. The reboot performs a stop for each service on that system and then reboots the operating system. Application failover and recovery in this case are the responsibility of the cluster-management software, which is activated as part of the reboot process.

Castelli at 318; *see id.* at 323.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1066. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at 317.

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. ***Events provide a notification mechanism from an agent into the IBM Director environment.***

Castelli at 315; *see also* 318, 323.

1067. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time that an agent or its runtime environment could be programmed to request a corrective policy from an entity outside of the particular runtime environment.

1068. To the extent this limitation is not met, it would have been obvious in view of Turek '070. See Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1069. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, "wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment." '659 Pat. at Cl. 1. For example:

A notification mechanism, via the built-in IBM Director's event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at 317.

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. ***Events provide a notification mechanism from an agent into the IBM Director environment.***

Castelli at 315; *see also* 318, 323.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1070. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

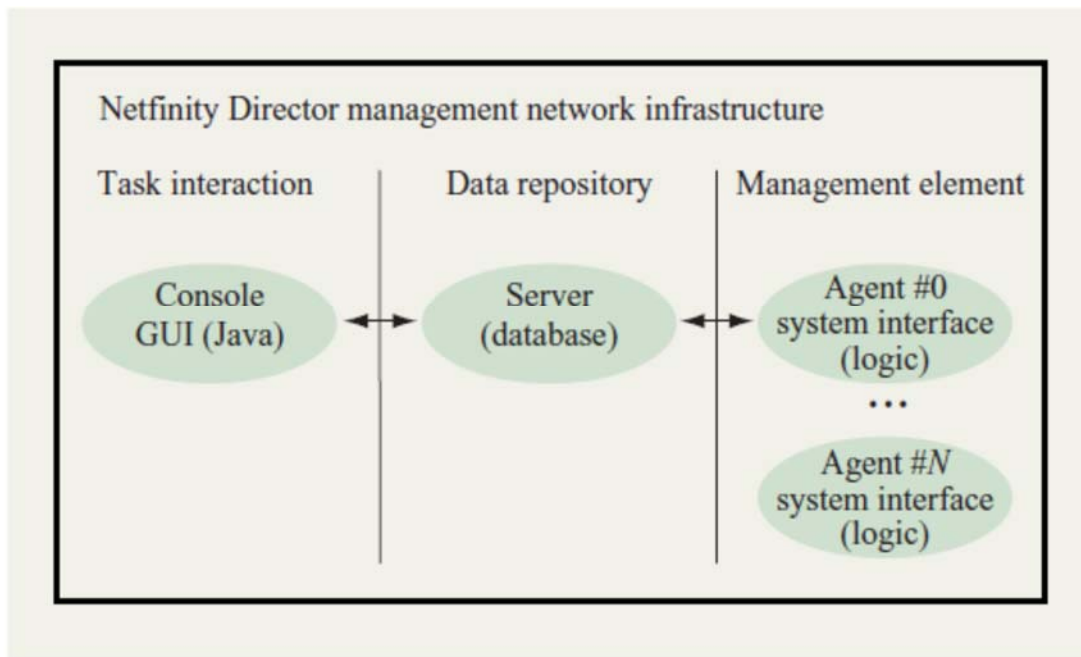
A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at 317.

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. ***Events provide a notification mechanism from an agent into the IBM Director environment.***

Castelli at 315; *see also* 318, 323.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1**

IBM Director framework.

Castelli at 315.

**2. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

1071. IBM xSeries Server Software Rejuvenation Agent discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

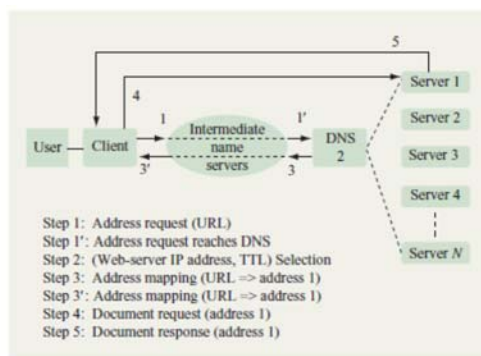
**IP dispatching**

As part of a web-hosting framework, an IP-dispatching or load-balancing component is often present to provide scalability, availability, and load-balancing capabilities for TCP/IP applications. Early implementations consisted of a domain name server (DNS) that would translate host names into IP addresses for corresponding servers, so that IP requests are routed in a round-robin fashion to a pool of servers. This approach is often called round-robin DNS; an example configuration is shown in Figure 16. Note that IP dispatching can be considered a specialized form of clustering, such that one or more nodes execute the dispatching components and the remaining nodes execute web-serving applications.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

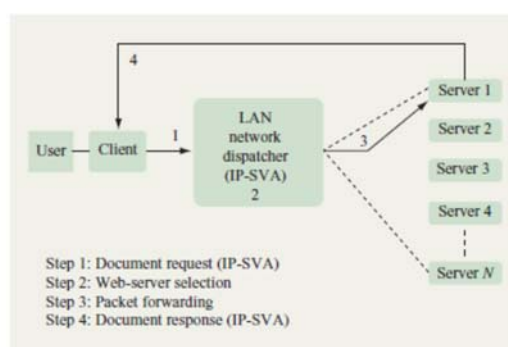
More advanced approaches are now available, such as the IBM Secureway Network Dispatcher [32]. This product provides enhanced IP-level load-balancing mechanisms and content-based routing, as well as improved management and availability functions. Figure 17 illustrates the network dispatcher operations for a LAN implementation. As part of load balancing, the dispatcher's scheduling policy is dynamically based on each server's load and availability. This is partly accomplished by having each server send periodic utilization information to the dispatcher. This utilization information can easily be augmented by health information in the form of time until resource exhaustion, degree of resource exhaustion or, in its simplest form, time remaining until a timed rejuvenation. This health information can be sent to the dispatcher in order to schedule actions for individual servers. The scheduling of these actions can also take into account aggregate loading of the web host.

Castelli at 323.



**Figure 16**

Example configuration of round-robin DNS.



**Figure 17**

Network dispatcher for a LAN implementation.

Castelli at 323

### 3. *Claim 3*

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1072. IBM xSeries Server Software Rejuvenation Agent discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

#### Prediction algorithms

In the current version of SRA, rejuvenation can be based on elapsed time since the last rejuvenation, or on prediction of impending exhaustion.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

When using timed rejuvenation, the user interface of IBM Director is used to schedule and perform rejuvenation at a period specified by the user. A calendar interface allows the user to select when to rejuvenate different nodes of the cluster, and to select "blackout" times during which no rejuvenation is to be allowed. Although this sounds rather primitive, our analysis (presented below) shows that for typical clusters that undergo aging, system availability can be improved significantly via this technique.

Castelli at 317.

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day's worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, *see also id.* at 323.

1073. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Algorithm.*

1074. IBM xSeries Server Software Rejuvenation Agent discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example:

***Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.*** The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the ***agent automatically performs the analysis.***

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and ***the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function*** without overfitting the data.

The ***prediction algorithm fits several types of curves to the data in the fitting window***; these curves have been selected for their ability to capture different types of temporal trends. ***A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon.*** Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, see also id. 324, 27, 28, 29.

1075. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

5. ***Claim 7***

- a. ***A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes***

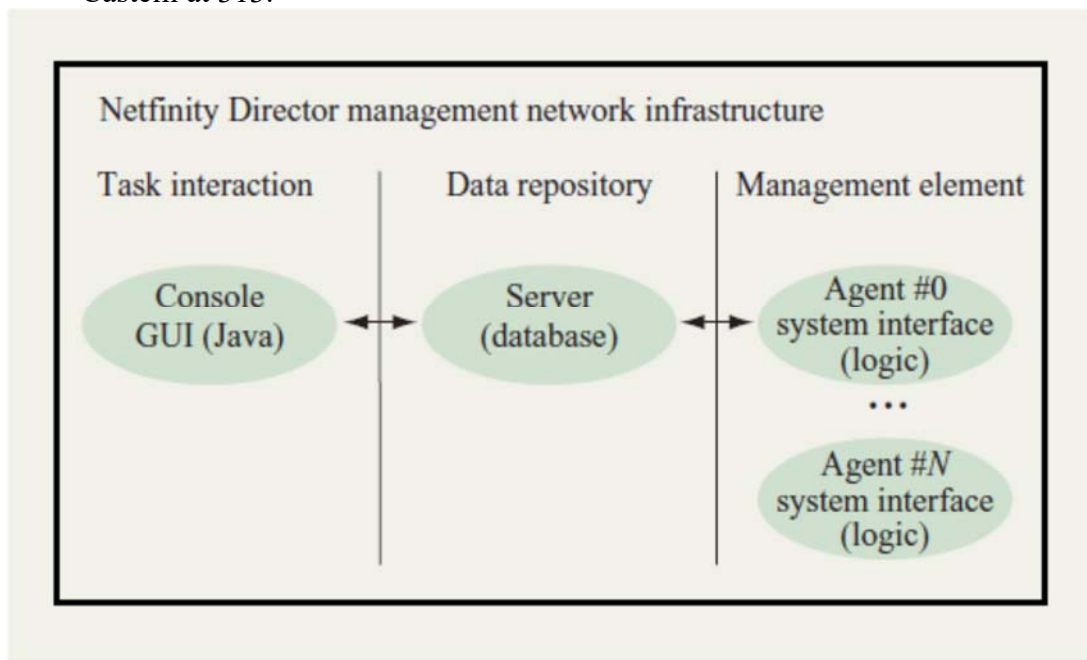
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

1076. IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

Castelli at 313.



**Figure 1**

**IBM Director framework.**

Castelli at 315; *see id.* at 312.

b. *running at least one thread in a first runtime environment;*

1077. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1078. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

1079. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- e. *and performing a corrective action to fix any detected abnormalities;*

1080. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1081. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1082. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

1083. IBM xSeries Server Software Rejuvenation Agent discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1084. IBM xSeries Server Software Rejuvenation Agent discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

1085. IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

This technology has been incorporated into the IBM Director for xSeries servers.

Castelli at 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castelli at 323.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

1086. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

This technology has been incorporated into the IBM Director for xSeries servers.

Castelli at 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castellit at 323.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1087. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

This technology has been incorporated into the IBM Director for xSeries servers.

Castellis at 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castelli at 323.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *running at least one thread in a first runtime environment;*

1088. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1089. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

- f. *detecting there is an abnormality in the monitored operational parameters;*

1090. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- g. *and performing a corrective action to fix any detected abnormalities;*

1091. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1092. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective policy applied by the agent running within the first runtime environment,*

1093. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1094. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1095. IBM xSeries Server Software Rejuvenation Agent discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1096. IBM xSeries Server Software Rejuvenation Agent discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

1097. IBM xSeries Server Software Rejuvenation Agent discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

1098. I expect to testify that IBM xSeries Server Software Rejuvenation Agent anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that IBM xSeries Server Software Rejuvenation Agent discloses each and every limitation of those claims is included as Ex. B-12.

**W. Anticipation by and/or Obviousness in View of United States Patent No. 6,826,150, filed 10/2/2001 and issued to Bhattacharya, et al. on 11/30/2004.**

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

1099. Bhattacharya discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

*A method for policing traffic on a computer* communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

1100. Further examples are also found at Bhattacharya at 1:14–17.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

1101. Bhattacharya discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

1102. Further examples include Bhattacharya at 3:21–44, 3:45–61, 3:62–4:10, 4:11–20, 4:56–5:22, 5:23–32, 5:33–53, 6:33–7:57; Bhattacharya at Figs. 1–3, 5.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1103. Bhattacharya discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl.

1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

1104. Further examples include Bhattacharya at 1:15–18, 2:57–3:4, 3:21–44, 3:45–61, 3:62–4:10, 4:11–20, 4:56–5:22, 5:23–32, 5:33–53, 6:33–7:57; Bhattacharya at Figs. 1–3, 5.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *detecting if there is an abnormality in the monitored operational parameters;*

1105. Bhattacharya discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

QoS is a feature that allows dropping of packets on a selective basis to avoid or reduce congestion in the network. Two components of QoS are "classification" and "policing." Packets are classified into different traffic classes according to policy set by the network administrator. For each class, a policing algorithm is used to measure the incoming traffic and compare that measure with policing parameters set by the network-administrator. ***As a result of policing, depending on the current traffic-rate for this class of traffic, a packet may be found “in profile” or “out of profile” by the policing algorithm. An out of profile packet is dropped or marked.*** Marking increases the probability of the packets being dropped later by another device that applies QoS to the packet. A packet that is dropped or marked by the policing algorithm is referred to as a “policed” packet. An in profile packet is forwarded without marking and is referred to as a packet “permitted” by the policing algorithm.

Bhattacharya at at 2:5–22.

1106. Further examples include Bhattacharya at 2:23–56, 3:21–44, 3:45–61, 5:33–53, 5:53–6:33, 6:33–7:57, 9:20–60; Bhattacharya at Figs. 3, 4, 6.

- e. *and performing a corrective action to fix any detected abnormalities,*

1107. Bhattacharya discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

1108. Further examples include Bhattacharya at 2:5–22, 3:45–61, 5:33–53, 5:53–6:33, 9:20–60; Bhattacharya at Figs. 3, 4, 6.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1109. Bhattacharya discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

QoS is a feature that allows dropping of packets on a selective basis to avoid or reduce congestion in the network. Two components of QoS are "classification" and "policing." Packets are classified into different traffic classes according to policy set by the network administrator. For each class, a policing algorithm is used to measure the incoming traffic and compare that measure with policing parameters set by the network-administrator. As a result of policing, depending on the current traffic-rate for this class of traffic, a packet may be found “in profile” or “out of profile” by the policing algorithm. An out of profile packet is dropped or marked. Marking increases the probability of the packets being dropped later by another device that applies QoS to the packet. A packet that is dropped or marked by the policing algorithm is referred to as a “policed” packet. An in profile packet is forwarded without marking and is referred to as a packet “permitted” by the policing algorithm.”

Bhattacharya at at 2:5–22.

1110. Further examples include Bhattacharya at 3:45–61, 5:33–53, 5:53–6:33, 9:20–60; Bhattacharya at Figs. 3, 4, 6.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1111. Bhattacharya discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

QoS is a feature that allows dropping of packets on a selective basis to avoid or reduce congestion in the network. Two components of QoS are "classification" and "policing." Packets are classified into different traffic classes according to policy set by the network administrator. For each class, a policing algorithm is used to measure the incoming traffic and compare that measure with policing parameters set by the network-administrator. As a result of policing, depending on the current traffic-rate for this class of traffic, a packet may be found “in profile” or “out of profile” by the policing algorithm. An out of profile packet is dropped or marked. Marking increases

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the probability of the packets being dropped later by another device that applies QoS to the packet. A packet that is dropped or marked by the policing algorithm is referred to as a “policed” packet. An in profile packet is forwarded without marking and is referred to as a packet “permitted” by the policing algorithm.

Bhattacharya at at 2:5–22.

1112. Further examples include Bhattacharya at 3:45–61, 5:33–53, 9:20–60; Bhattacharya at Figs. 3, 6.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1113. Bhattacharya discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

The present invention provides individual policers that monitor and police part of the traffic. In order to accomplish this policing by individual policers the contract rate and the burst for an entire traffic class is provided to the individual policers. The individual policers measure their parts of the traffic and export that information to all the other individual policers. There is a mechanism for receiving, totalizing and storing the individual policer information exchanged from all the individual policers. That total is then compared to the contract rate and burst for the entire traffic class and a policing decision is made by each individual policer for its part of the traffic class. In an embodiment, there is a master policer that receives all the information from all the individual policers and applies the contract rate and the burst and makes policing decisions for the individual policers that is then sent back to the individual policers.

Bhattacharya at 3:45-61.

1114. Further examples include Bhattacharya at Abstract, 3:62–4:10, 4:11–20, 4:21–38, 4:56–5:22, 5:23–32, 5:33–53, 6:33–7:57, Cl. 2.; Bhattacharya at Figs. 1–3, 5.

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

1115. Bhattacharya discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

1116. Further examples include Bhattacharya at 2:5–22, 3:45–61, 5:53–6:33; Bhattacharya at Fig. 4.

**3. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1117. Bhattacharya discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bhattacharya at Abstract.

1118. Further examples include Bhattacharya at 2:5–22, 2:23–56, 3:45–61, 5:33–53; Bhattacharya at Fig. 3.

1119. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

**4. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

1120. Bhattacharya discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.” '659 Pat. at Cl. 6. For example:

The inventive system and method implements the policing policy across the entire network, or part thereof, by providing many modules referenced as individual policers. Each individual policer can observe and police only a part of the traffic. In this system each individual policer uses “global state variables” that stores the measure traffic for the entire traffic-class and “local state variables” that store the measure of the part of traffic that is permitted by this individual policer. Each individual policer exports the measure of traffic it permitted to all individual policers, in the form of the local state variables, or functions thereof. After exporting such information, the individual policer clears the local state variables. Upon receiving such exported information, all the individual policers update their global state variables. At any time, the global state variables of an individual policer account for the total of all the measure of traffic exported from all individual policers till that time. At any time, the local state variables of an individual policer account for the traffic that was permitted at the same individual policer since it last exported the measure of traffic it permitted.

Bhattacharya at 3:21-44.

1121. Further examples include Bhattacharya at 3:45–61.

1122. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

**5. Claim 7**

- a. *A non-transitory computer-readable medium comprising a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

1123. Bhattacharya discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

Various algorithms exist to perform policing. Each of these algorithms is implemented in a single logic-module called policer that performs the same computation for all packets belonging to a traffic class. Such a policer is referenced as a centralized policer, since the same module needs to perform the computation for all packets belonging to a traffic class. One such policing algorithm limits the total number bytes in all packets permitted in any arbitrary time-interval, T, to the value of  $(T * \text{contract\_rate} + \text{burst})$ . Here and as defined below, “contract rate” is a policing policy parameter meaning information per unit time, and “burst” is another policing policy parameter meaning the maximum information permitted in excess of the rate. This can be implemented in a policer called the token-bucket policer, which performs the following computation for every packet in a traffic class

Bhattacharya at 2:23-56.

1124. Further examples include Bhattacharya 2:57–3:4, 3:5–17, 4:11–20, 4:21–38, 5:53–6:33, 6:33–7:57, 9:20–60; Bhattacharya at Figs. 4–6.

b. *running at least one thread in a first runtime environment;*

1125. Bhattacharya discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1126. Bhattacharya discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

1127. Bhattacharya discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

1128. Bhattacharya discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1129. Bhattacharya discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1130. Bhattacharya discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

1131. Bhattacharya discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1132. Bhattacharya discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

a. *A system, comprising:*

1133. Bhattacharya discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

A centralized policer requires all packets to be processed for policing in a single logic-module. So, packets that are processed in different logic modules cannot belong to the same traffic-class. An object of the present invention is to allow packets arriving at multiple logic-modules to be policed as a single traffic-class. Similarly, an associated object of the present invention is to allow packets arriving at multiple network devices to be policed as a single traffic class. The invention requires each such logic-module to send information about packets that the logic module processed to other such logic-modules. There is a cost associated with the communication capacity used for exchanging such information. If such “overhead” information is sent less often, it uses less of the capacity of the communication system, but it also decreases the accuracy of policing.

Bhattacharya at 2:57–3:4.

1134. Further examples include Bhattacharya at 3:21–44, 3:45–61, 4:56–5:22; Bhattacharya at Fig. 1.

b. *a processor;*

1135. Bhattacharya discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

A centralized policer requires all packets to be processed for policing in a single logic-module. So, packets that are processed in different logic modules cannot belong to the same traffic-class. An object of the present invention is to allow packets arriving at multiple logic-modules to be policed as a single traffic-class. Similarly, an associated object of the present invention is to allow packets arriving at multiple network devices to be policed as a single traffic class. The invention requires each such logic-module to send information about packets that the logic module processed to other such logic-modules. There is a cost associated with the communication capacity used for exchanging such information. If such “overhead” information is sent less often, it uses less of the capacity of the communication system, but it also decreases the accuracy of policing.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bhattacharya at 2:57–3:4.

1136. Further examples include Bhattacharya at 3:45–61, 5:33–53; Bhattacharya at Fig.

3.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1137. Bhattacharya discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

A centralized policer requires all packets to be processed for policing in a single logic-module. So, packets that are processed in different logic modules cannot belong to the same traffic-class. An object of the present invention is to allow packets arriving at multiple logic-modules to be policed as a single traffic-class. Similarly, an associated object of the present invention is to allow packets arriving at multiple network devices to be policed as a single traffic class. The invention requires each such logic-module to send information about packets that the logic module processed to other such logic-modules. There is a cost associated with the communication capacity used for exchanging such information. If such “overhead” information is sent less often, it uses less of the capacity of the communication system, but it also decreases the accuracy of policing.

Bhattacharya at 2:57–3:4.

1138. Further examples include Bhattacharya at 3:5–17, 3:21–44, 3:45–61, 4:11–20, 4:21–38, 5:23–32, 5:53–6:33; Bhattacharya at Fig. 4.

- d. *running at least one thread in a first runtime environment;*

1139. Bhattacharya discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1140. Bhattacharya discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

- f. *detecting there is an abnormality in the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

1141. Bhattacharya discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

1142. Bhattacharya discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1143. Bhattacharya discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1144. Bhattacharya discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1145. Bhattacharya discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1146. Bhattacharya discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1147. Bhattacharya discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1148. Bhattacharya discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

1149. I expect to testify that this art anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

**X. Anticipation by and/or Obviousness in View of United States Patent No.**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6,219,719, filed on 1/22/1997 and issued to Graf on 4/17/2001.

1. *Claim 1*

a. *A computer-implemented method, comprising:*

1150. Graf discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:

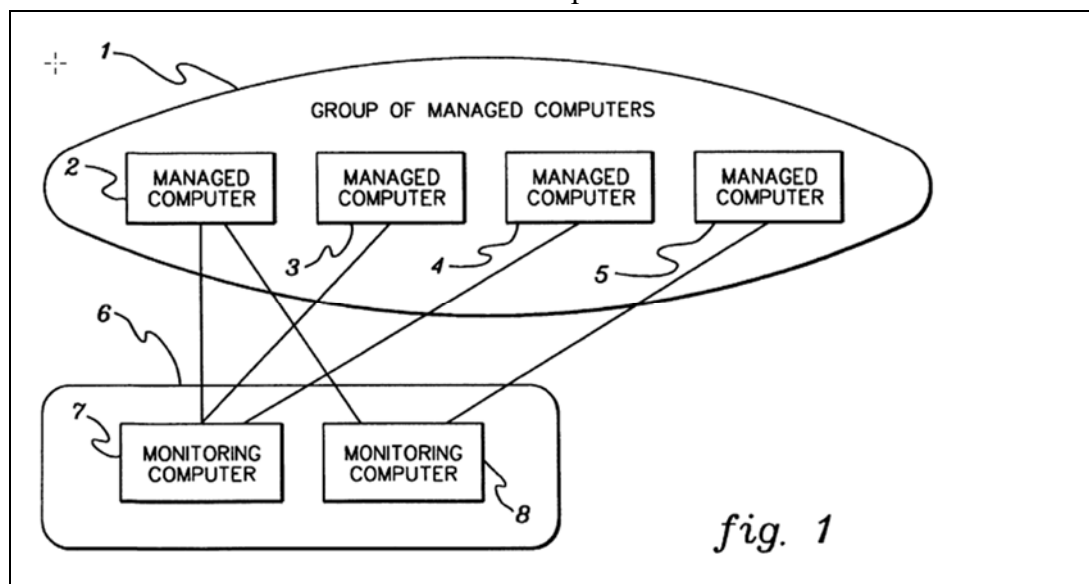
The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

1151. Further examples are also found at Graf at 3:14–22, 3:24–34, 3:35–38, 4:14–42, 4:43–5:8, 5:10–37, 8:14–22, 12:54–13:9, 39:30–36; Graf at Figs. 1, 2.

b. *running at least one thread in a first runtime environment;*

1152. Graf discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:





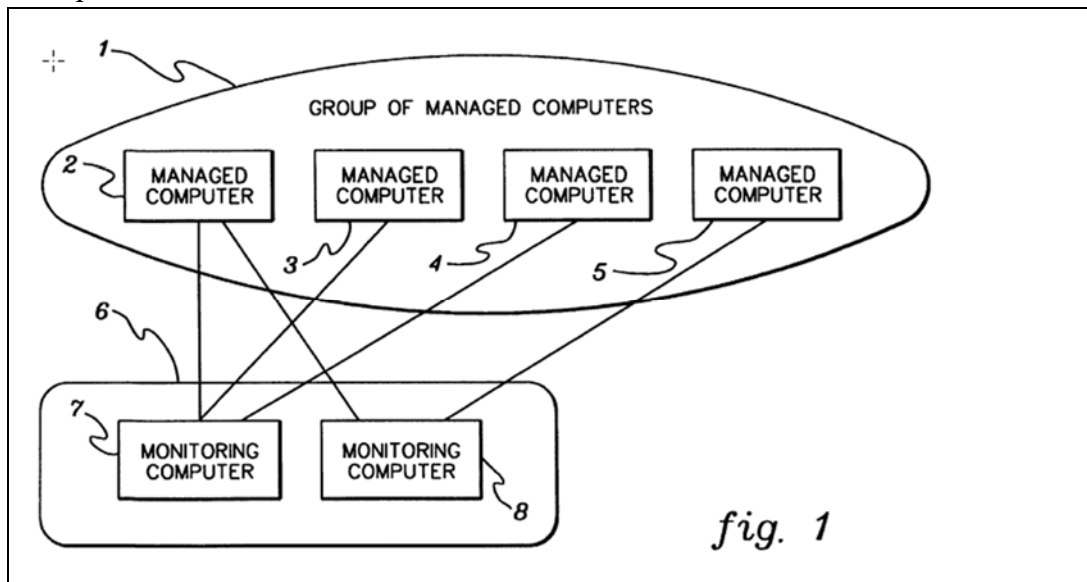
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1153. Graf explains, “FIG. 1 illustrates an embodiment of the present invention which comprises two groups of computers, a group of managed computers and a group of monitoring computers.” Graf at 3:32–34.

1154. Further examples include Graf at 4:14–42, 4:43–5:8, 5:10–37, 6:28–7:6, 7:38–8:2, 8:36–43, 9:57–10:4, 10:25–39, 12:54–13:9, 15:48–61, 37:54–38:65; Graf at Fig. 4.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1155. Graf discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:



1156. Graf explains, “FIG. 1 illustrates an embodiment of the present invention which comprises two groups of computers, a group of managed computers and a group of monitoring computers.” Graf at 3:32–34.

1157. Further examples include Graf at 4:14–42, 4:43–5:8, 8:53–9:18, 9:25–48, 9:57–10:4, 10:25–39, 12:54–13:9, 14:13–39, 14:46–67, 15:1–20, 17:21–53; Graf at Fig. 4.

- d. *detecting if there is an abnormality in the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

1158. Graf discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

1159. Further examples include Graf at 4:43–5:8, 8:36–43, 9:57–10:4, 12:54–13:9, 14:13–39, 17:21–53, 21:52–66, 23:25–66, 27:1–20, 37:54–38:65, 40:34–47, 41:28–47.

e. *and performing a corrective action to fix any detected abnormalities,*

1160. Graf discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example: “What is needed is a tool which will automatically gather the necessary computer information to manage a group of computers, detect problems based upon the gathered information, inform the system administrator of detected problems, and ***automatically perform corrective actions to resolve detected problems.***” Graf at 3:6–11.

1161. Further examples include Graf at 3:14–22, 4:43–5:8, 5:10–37, 8:36–43, 9:57–10:4, 12:54–13:9, 14:13–39, 17:54–18:25, 18:26–62, 19:24–36, 19:36–20:17, 20:21–60, 20:62–21:41, 21:52–66, 23:25–66, 37:54–38:65, 40:34–47, 41:28–47.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1162. Graf discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In another embodiment the two groups of computers may be the same group (all managed computers are also monitoring computers), two distinct groups (no managed computers are monitoring computers), or overlap (some managed computers are monitoring computers). The computers which form the groups of computers may be heterogeneous or homogeneous. The only requirement is that each managed computer have the capability to communicate with at least one monitoring computer. One preferred embodiment of this invention is to have all the computers on a computer network, but any other means of communication, e.g., over a modem using a telecommunications network, is adequate. The differentiation between managed and monitoring computers are the SYSTEMWatch AI-L client and the SYSTEMWatch AI-L console, which are described below:

a. As show in FIG. 2, a computer is a managed computer if the computer is running the SYSTEMWatch AI-L client, which provides a means for the computer to automatically detect and respond to problems. Additionally, the SYSTEMWatch AI-L client also accepts and responds to commands issued by a SYSTEMWatch AI-L console described below.

b. As shown in FIG. 3, a computer is a monitoring computer if the computer is running the SYSTEMWatch AI-L console, which provides a means for the computer to receive and display notifications of detected problems, and to display the corrective actions taken. Additionally, the SYSTEMWatch AI-L console is also able to issue commands to any group of managed computers.

c. As shown in FIG. 4, a computer is both a managed computer and a monitoring computer if it contains both SYSTEMWatch AI-L client, 13, and SYSTEMWatch AI-L console, 21.

Graf at 4:43–5:8.

1163. Further examples include at Graf 3:6–11, 5:10–37, 44:19–24, 8:36–43, 9:57–10:4, 12:54–13:9, 14:13–39, 17:54–18:25, 19:36–20:17, 20:21–60, 20:62–21:41, 21:52–66, 23:25–66, 27:1–20, 37:54–38:65, 40:34–47, 41:28–47, 41:47–66, 42:29–65.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1164. Graf discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

In another embodiment the two groups of computers may be the same group (all managed computers are also monitoring computers), two distinct groups (no managed computers are monitoring computers), or overlap (some managed computers are monitoring computers). The computers which form the groups of computers may be heterogeneous or homogeneous. The only requirement is that each managed computer have the capability to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

communicate with at least one monitoring computer. One preferred embodiment of this invention is to have all the computers on a computer network, but any other means of communication, e.g., over a modem using a telecommunications network, is adequate. The differentiation between managed and monitoring computers are the SYSTEMWatch AI-L client and the SYSTEMWatch AI-L console, which are described below:

a. As show in FIG. 2, a computer is a managed computer if the computer is running the SYSTEMWatch AI-L client, which provides a means for the computer to automatically detect and respond to problems. Additionally, the SYSTEMWatch AI-L client also accepts and responds to commands issued by a SYSTEMWatch AI-L console described below.

b. As shown in FIG. 3, a computer is a monitoring computer if the computer is running the SYSTEMWatch AI-L console, which provides a means for the computer to receive and display notifications of detected problems, and to display the corrective actions taken. Additionally, the SYSTEMWatch AI-L console is also able to issue commands to any group of managed computers.

c. As shown in FIG. 4, a computer is both a managed computer and a monitoring computer if it contains both SYSTEMWatch AI-L client, 13, and SYSTEMWatch AI-L console, 21.

Graf at 4:43–5:8.

1165. Further examples include Graf at 5:10–37, 9:57–10:4, 17:21–53, 17:54–18:25, 20:21–60, 20:62–21:41, 37:54–38:65, 41:47–66, 42:29–65, 41:47–66, 42:29–65.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1166. Graf discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

In another embodiment the two groups of computers may be the same group (all managed computers are also monitoring computers), two distinct groups (no managed computers are monitoring computers), or overlap (some managed computers are monitoring computers). The computers which form the groups of computers may be heterogeneous or homogeneous. The only requirement is that each managed computer have the capability to communicate with at least one monitoring computer. One preferred embodiment of this invention is to have all the computers on a computer network, but any other means of communication, e.g., over a modem using a telecommunications network, is adequate. The differentiation between

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

managed and monitoring computers are the SYSTEMWatch AI-L client and the SYSTEMWatch AI-L console, which are described below:

a. As show in FIG. 2, a computer is a managed computer if the computer is running the SYSTEMWatch AI-L client, which provides a means for the computer to automatically detect and respond to problems. Additionally, the SYSTEMWatch AI-L client also accepts and responds to commands issued by a SYSTEMWatch AI-L console described below.

b. As shown in FIG. 3, a computer is a monitoring computer if the computer is running the SYSTEMWatch AI-L console, which provides a means for the computer to receive and display notifications of detected problems, and to display the corrective actions taken. Additionally, the SYSTEMWatch AI-L console is also able to issue commands to any group of managed computers.

c. As shown in FIG. 4, a computer is both a managed computer and a monitoring computer if it contains both SYSTEMWatch AI-L client, 13, and SYSTEMWatch AI-L console, 21.

Graf at 4:43–5:8.

1167. Further examples include Graf at 9:57–10:4, 14:13–39, 17:21–53, 40:34–47.

2. ***Claim 2***

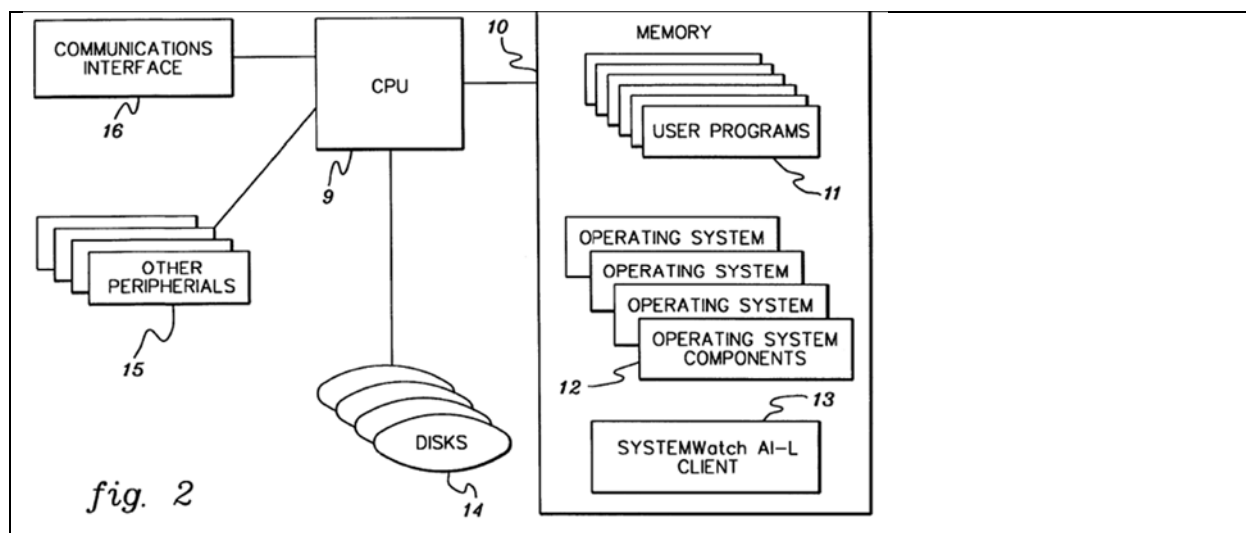
a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

1168. Graf discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

1169. Graf’s Fig. 2 is illustrative, “FIG. 2 illustrates one example of the structure of a managed computer, comprising a processing unit, memory, disk, network interface, peripherals, and a SYSTEMWatch AI-L client.” Graf at 3:35-38.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1170. Further examples include Graf at 3:35–38, 5:10–37, 6:28–7:6, 8:14–22, 12:54–13:9, 33:49–66, 36:1–12.

3. **Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1171. Graf discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

Core Layer Description—The Expert System

The second element of the core layer is an expert system, 40, which is used for problem detection and action initiation. The expert system, 40, is a forward chaining rule based expert system using a rule specificity algorithm. When SYSTEMWatch AI-L client, 13, is started, the expert system contains no rules. Rules are declared and incorporated into the core layer. Rules support both the IF-THEN rules as well as IF-THEN-ELSE rules. The rules used in SYSTEMWatch AI-L permit assignments and function calls within the condition of the rule. Additionally, SYSTEMWatch AI-L expert system, 40, also has the following features:

- a. Rules can declare variables. All variables declared within a rule are static variables.
- b. Rules can have an initialization section. The initialization section contains actions which must be performed only once, and before the rule is ever tested. It can, for example, contain a state declaration and an interval declaration (states and intervals are described below). It may contain

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

variable declarations for variables used by the rules, and it may contain code to do a variety of actions.

c. Rules can have, for instance, an INTERVAL and a LASTCHECK time. In accordance with the principles of the present invention, in order for a rule to be eligible for testing by the expert system, at the time of testing the clock time must be equal to or greater than the LASTCHECK time plus the INTERVAL time. The LASTCHECK time for each rule is set to the clock time whenever a rule is actually tested. This way, the INTERVAL specifies the minimum amount of time which must elapse since the last time a rule was checked before the rule becomes eligible for testing again.

Graf at 7:38–8:2.

1172. Further examples include Graf at 12:54–13:9, 14:13–39.

1173. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

1174. Graf discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

1175. Further examples include Graf at 8:36–43, 33:6–31.

1176. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**5. *Claim 7*

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

1177. Graf discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example:

a. A group of managed computers, **1**, which includes computers, **2-5**, comprising, for example, (see FIG. 2) a CPU, **9**, memory, **10**, disks, **14**, communications interface, **16**, other peripherals, **15**, and a SYSTEMWatch AI-L client, **13**. The size of the managed group of computers can range from 1 to several thousand. Data which is gathered from a managed computer is stored on the managed computer. From time to time, a managed computer may send data to a monitoring computer (see below).

b. A group of monitoring computers, **6**, which includes computers comprising, for example, (see FIG. 3) a CPU, **17**, memory, **18**, disks, **22**, communications interface, **24**, other peripherals, **23**, and a SYSTEMWatch AI-L console, **21**. The size of the monitoring group of computers can range from 0 to several hundred. Although data gathered from a managed computer is stored on the managed computer, from time to time a managed computer may send data to a monitoring computer. A monitoring computer can also explicitly request data from a managed computer. Data which is received by the monitoring computer from a managed computer is stored on the monitoring computer. Furthermore, since a monitoring computer can receive data from several managed computers, a monitoring computer may perform post-processing on data received from several managed computer, and/or perform additional data gathering itself, in which case that data is stored on the monitoring computer.

Graf at 4:14–42.

1178. Further examples include Graf at 5:10–37, 6:28–7:6, 7:7–37, 7:38–8:2, 8:14–22, 8:53–9:18, 9:25–48, 9:57–10:4, 10:25–39, 12:54–13:9, 14:13–39, 15:48–61, 17:21–53, 37:54–38:65, 45:1–5, 46:27–37, 47:5–42.

- b. *running at least one thread in a first runtime environment;*

1179. Graf discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

1180. Graf discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

1181. Graf discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *performing a corrective action to fix any detected abnormalities;*

1182. Graf discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1183. Graf discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1184. Graf discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

1185. Graf discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1186. Graf discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. ***Claim 13***

- a. *A system, comprising:*

1187. Graf discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

1188. Further examples include Graf at 3:14–22, 3:32–34, 3:35–38, 3:44–48, 4:14–42, 5:10–37, 6:28–7:6, 7:38–8:2, 12:54–13:9, 17:21–53, 32:26–34, 40:34–47; Graf at Figs. 1, 2, 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

1189. Graf discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

1190. Further examples include Graf at 3:14–22, 3:35–38, 3:44–48, 4:14–42, 4:43–5:8, 5:10–37, 6:28–7:6, 7:38–8:2, 8:14–22, 17:21–53, 32:26–34; Graf at Figs. 1, 4.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1191. Graf discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1192. Further examples include Graf at 3:14–22, 3:35–38, 4:14–42, 4:43–5:8, 5:10–37, 8:14–22, 8:53–9:18, 9:25–48, 9:57–10:4, 12:54–13:9, 14:46–67, 15:48–61, 17:21–53, 32:26–34, 37:54–38:65; Graf at Fig. 1.

d. *running at least one thread in a first runtime environment;*

1193. Graf discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1194. Graf discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

1195. Graf discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

1196. Graf discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1197. Graf discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1198. Graf discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1199. Graf discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**9. Claim 14**

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1200. Graf discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**10. Claim 15**

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1201. Graf discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

1202. Graf discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

1203. I expect to testify that this art anticipates each Asserted Claim of the '659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

**Y. Anticipation by and/or Obviousness in View of United States Patent No. 6,243,667, filed 5/28/1996 and issued to Kerr, et al. on 6/5/2001.**

1204. United States Patent No. 6,243,667 was filed on May 28, 1996 and issued to Darren R. Kerr and Barry L. Bruins on June 5, 2001 (“Kerr”). The patent describes switching in networks responsive to message flow patterns. *See, e.g.*, Kerr at abstract.

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

1205. Kerr discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example, “embodiments of this invention may be ***implemented using a set of general purpose computers*** operating under program control.” Kerr at 2:31–33.

1206. Further examples include Kerr at 1:48–49; Kerr at 2:14–15; Kerr at 2:21–22; Kerr at Cl. 16.

- b. *running at least one thread in a first runtime environment;*

1207. Kerr discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example, “embodiments of this invention may be implemented using a set of general purpose computers operating under program control.” Kerr at 2:31–33.

1208. Further examples include Kerr at 1:49–55; Kerr at 2:56–3:2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1209. Kerr discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

1210. Further examples include Kerr at 5:37–48; Kerr at 3:40–45; Kerr at 6:32–41; Kerr at 8:50–9:10; Kerr at Cl. 16.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

1211. Kerr discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

1212. Further examples include Kerr at 8:50–9:10; Kerr at 5:37–48.

- e. *and performing a corrective action to fix any detected abnormalities,*

1213. Kerr discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

1214. Further examples include Kerr at 4:20–33; Kerr at 5:17–25; Kerr at 5:37–48; Kerr at 8:50–9:10.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1215. Kerr discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

At a step 532, the reporting device 540 builds a report about a condition of the network 100, responsive to information about message flows 160.

At a step 533, the reporting device 540 displays or transmits that report about the condition of the network 100 to interested parties.

Kerr at 8:60–65.

1216. Kerr then continues:

Interested parties may diagnose actual or potential network problems. For example, the report may comprise information about packets 150 in particular message flows 160, including a time stamp for a first packet 150 and a time stamp for a last packet 150 in the message flow 160, a cumulative total number of bytes in the message flow 160, a cumulative total number of packets 150 in the message flow 160, or other information relevant to diagnosing actual or potential network problems.

Kerr at 8:60–9:10.

1217. Further examples include Kerr at 6:50–64.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

1218. Kerr discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Because the routing device 140 processes each packet 150 in the message flow 160 responsive to the entry for the message flow 160 in the flow cache, the routing device 140 is able to implement administrative policies which are designated for each message flow 160 rather than for each packet 150. For example, the routing device 140 is able to reserve specific amounts of bandwidth for particular message flows 160 and to queue packets 150 for transmission responsive to the bandwidth reserved for their particular message flows 160.

Kerr at 5:17–25.

1219. Further examples include Kerr at 6:50–64; Kerr at Cl. 16.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1220. Kerr discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

At a step 532, the reporting device 540 builds a report about a condition of the network 100, responsive to information about message flows 160.

At a step 533, the reporting device 540 displays or transmits that report about the condition of the network 100 to interested parties.

Kerr at 8:60–65.

1221. Kerr then continues:

Interested parties may diagnose actual or potential network problems. For example, the report may comprise information about packets 150 in particular message flows 160, including a time stamp for a first packet 150 and a time stamp for a last packet 150 in the message flow 160, a cumulative total number of bytes in the message flow 160, a cumulative total number of packets 150 in the message flow 160, or other information relevant to diagnosing actual or potential network problems.

Kerr at 8:60–9:10.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

1222. Kerr discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

1223. Further examples include Kerr at 4:20–33.

3. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1224. Kerr discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

1225. Further examples include Kerr at 9:45–67.

4. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Algorithm.*

1226. Kerr discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example:

For a second example, the routing device 140 is able to monitor usage information regarding relative use of network resources, and to give priority to those message flows 160 which use relatively fewer network resources. This can occur when a first message flow 160 is using a relatively low-bandwidth transmission channel (such as a 28.8 kilobits per second modem transmission channel) and when a second message flow 160 is using a relatively high-bandwidth transmission channel (such as a T-1 transmission line).

Kerr at 5:37–48.

1227. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

**5. Claim 7**

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

1228. Kerr discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. For example, “embodiments of this invention may be implemented using a set of general purpose computers operating under program control.” Kerr at 2:31–33.

- b. *running at least one thread in a first runtime environment;*

1229. Kerr discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b.

- c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

1230. Kerr discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

1231. Kerr discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

e. *performing a corrective action to fix any detected abnormalities;*

1232. Kerr discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1233. Kerr discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1234. Kerr discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

1235. Kerr discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1236. Kerr discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

1237. Further examples include

8. ***Claim 13***

- a. *A system, comprising:*

1238. Kerr discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example:

The invention provides a method and system for Switching in networks responsive to message flow patterns. A message "flow" is defined to comprise a set of packets to be transmitted between a particular Source and a particular destination. When routers in a network identify a new message flow, they determine the proper processing for packets in that message flow and cache that information for that message flow. Thereafter, when routers in a network identify a packet which is part of that message flow, they process that packet according to the proper processing for packets in that message flow. The proper processing may include a determination of a destination port for routing those packets and a determination of whether access control permits routing those packets to their indicated destination.

Kerr at 1:48–61

1239. Further examples include Kerr at Cl. 11.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

1240. Kerr discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example, “embodiments of this invention may be implemented using a set of general purpose computers operating under program control.” Kerr at 2:31–33.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1241. Kerr discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example, “embodiments of this invention may be implemented using a set of general purpose computers operating under program control.” Kerr ’667 at 2:31–33.

d. *running at least one thread in a first runtime environment;*

1242. Kerr discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1243. Kerr discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

1244. Kerr discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected abnormalities;*

1245. Kerr discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1246. Kerr discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1247. Kerr discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1248. Kerr discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1249. Kerr discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1250. Kerr discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1251. Kerr discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

1252. I expect to testify that this art anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

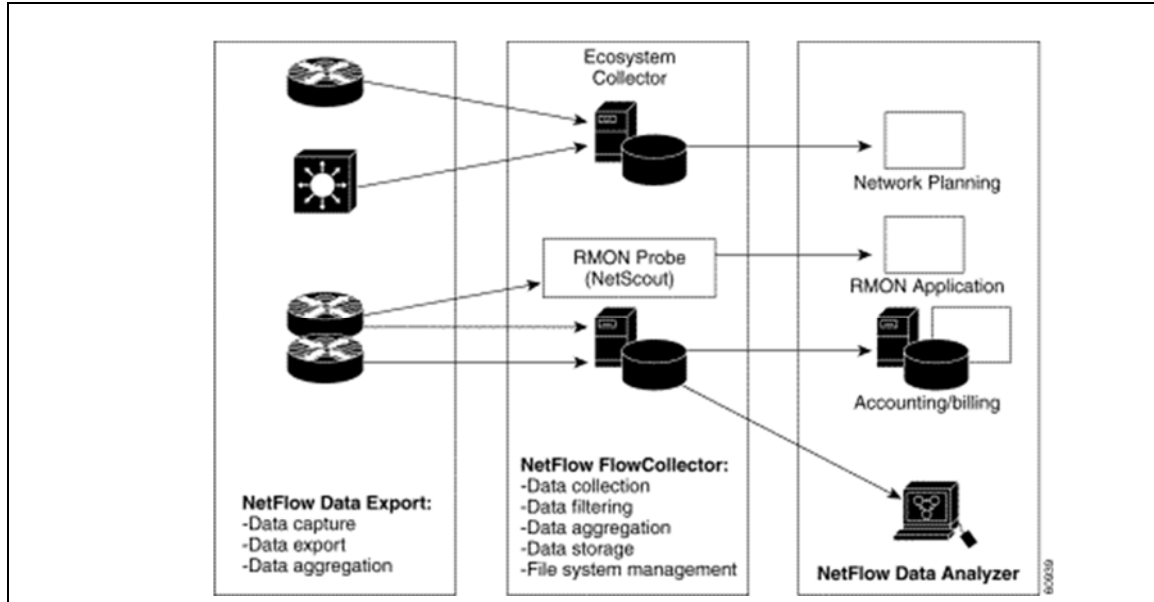
**Z. Anticipation by and/or Obviousness in View of Cisco NetFlow, marketed by Cisco by 1995 at the latest.**

1. ***Claim 1***

- a. *A computer-implemented method, comprising:*

1253. Cisco NetFlow discloses the preamble of claim 1, “[a] computer-implemented method, comprising . . . .” ’659 Pat. at Cl. 1. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1).

1254. Additionally, ““NetFlow is a Cisco IOS application that provides statistics on packets flowing through the routing devices in the network. It is emerging as a primary network accounting and security technology.” NetFlow Configuration Guide at 1.

b. *running at least one thread in a first runtime environment;*

1255. Cisco NetFlow discloses this element of claim 1, “running at least one thread in a first runtime environment.” ’659 Pat. at Cl. 1. For example:

NetFlow Data Export—Capture NetFlow accounting statistics for unicast ingress traffic or MPLS egress traffic on your networking device, and export your data to a collection device. Expired packet streams or flows are grouped together into "NetFlow Export" User Datagram Protocol (UDP) datagrams for export to a collection device. Using UDP datagrams, you provide simplicity and speed to your network compared to TCP networks where each packet is acknowledged. NetFlow allows you to aggregate NetFlow data on the routing device before exporting to a collection device, resulting in lower bandwidth requirements for NetFlow data and reduced platform requirements for NetFlow data collection devices.

NetFlow Solutions Guide at § NetFlow Infrastructure.

1256. Further examples include ISP Essentials at 44; NetFlow Configuration Guide at 2.

c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

1257. Cisco NetFlow discloses this element of claim 1, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat. at Cl. 1. For example: “Network monitoring—NetFlow data enables extensive near real-time network monitoring capabilities. You can use NetFlow flow data analysis to display traffic patterns associated with individual routing devices and switches as well as on a network-wide basis (providing aggregate traffic or application-based views) to provide proactive problem detection, efficient troubleshooting, and rapid problem resolution.” NetFlow Solutions Guide at § Why Deploy NetFlow?

1258. Further examples include NetFlow Solutions Guide at § Capturing NetFlow Data; ISP Essentials at 44; NetFlow Configuration Guide at 2; Cyber Ops at 78.

d. *detecting if there is an abnormality in the monitored operational parameters;*

1259. Cisco NetFlow discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. For example: “Network monitoring—NetFlow data enables extensive near real-time network monitoring capabilities. You can use NetFlow flow data analysis to display traffic patterns associated with individual routing devices and switches as well as on a network-wide basis (providing aggregate traffic or application-based views) to provide proactive problem detection, efficient troubleshooting, and rapid problem resolution.” NetFlow Solutions Guide at § Why Deploy NetFlow?

1260. Further examples include ISP Essentials at 44; NetFlow Configuration Guide at 2; Cyber Ops at 78.

e. *and performing a corrective action to fix any detected abnormalities,*

1261. Cisco NetFlow discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. For example: “Network monitoring—NetFlow data enables extensive near real-time network monitoring capabilities. You can use NetFlow flow data analysis to display traffic patterns associated with individual routing devices

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

and switches as well as on a network-wide basis (providing aggregate traffic or application-based views) to provide proactive problem detection, efficient troubleshooting, and rapid problem resolution.” NetFlow Solutions Guide at § Why Deploy NetFlow?

1262. Further examples include NetFlow Solutions Guide at § Network Data Analyzer.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1263. Cisco NetFlow discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

#### NetFlow Policy Routing

The NetFlow Policy Routing (NPR) feature integrates policy routing, which enables traffic engineering and traffic classification, with NetFlow Services. IP policy routing now works with CEF, dCEF, NetFlow, and NetFlow flow acceleration.

As long as policy routing is configured, NPR is enabled by default and cannot be disabled; that is, NPR is the default policy routing mode. No configuration is required to enable policy routing in conjunction with CEF, dCEF, or NetFlow.

#### NPR Benefits

As Quality of Service (QoS) and traffic engineering become more popular, so does interest in the ability of policy routing to selectively set IP precedence and ToS bits (based on ACLs and packet size), thereby routing packets based on predefined policy. It is important that policy routing work well in large, dynamic routing environments. Hence, distributed support allows you to leverage your investment in distributed architecture.

#### NPR—Configuration Example

The following configuration example shows how to configure NPR using either CEF, NetFlow, and NetFlow with flow acceleration; it also configures policy routing to verify that next hop 50.0.0.8 of route map named test is a CDP neighbor before the routing device tries to policy route to it.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

NetFlow Solutions Guide at § NetFlow Policy Routing.

NetFlow can capture data over a long period of time, which enables you to track and anticipate network growth and plan upgrades. NetFlow service data can be used to optimize network planning, which includes peering, backbone upgrade planning, and routing policy planning. It also enables you to minimize the total cost of network operations while maximizing network performance, capacity, and reliability. NetFlow detects unwanted WAN traffic, validates bandwidth and quality of service (QoS) usage, and enables the analysis of new network applications. NetFlow offers valuable information that you can use to reduce the cost of operating the network.

NetFlow Configuration Guide at 2.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1264. Cisco NetFlow discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. For example:

Network planning (resources tracking and capacity planning)—NetFlow data captured in long-term observations allows you to track and anticipate network growth and plan upgrades with additional routing devices, ports, or higher-bandwidth interfaces. NetFlow data used with tools such as Netsys optimize both strategic network planning (such as who to peer with, backbone upgrade planning, and routing policy planning) as well as tactical network engineering decisions (such as adding additional Versatile Interface Processors [VIPs] to routing devices, and upgrading link capacity) to minimize the total cost of network operations while maximizing network performance, capacity, and reliability.

NetFlow Solutions Guide at § Why Deploy NetFlow?

1265. Further examples include NetFlow Configuration Guide at 2.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1266. Cisco NetFlow discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

[NetFlow FlowCollector]—Provides scalable and economical data collection from multiple NetFlow-enabled devices. The NFC is a UNIX application supported on Solaris and HP-UX platforms. The NFC provides the following functionality:

Consumes flows from multiple NetFlow-enabled devices

Performs data volume reduction through selective filtering and aggregation

Stores flow information in flat files on disk for post-processing by NetFlow data consumers, third-party billing applications, and traffic analysis tools.

NetFlow Solutions Guide at § NetFlow Infrastructure.

1267. Further examples include NetFlow Solutions Guide at § Why Deploy NetFlow.

**2. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

1268. Cisco NetFlow discloses claim 2, “[t]he computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.” ’659 Pat. at Cl. 2. For example:

Traffic engineering—NetFlow provides autonomous system (AS) traffic engineering details for an AS. You can use NetFlow-captured traffic data to understand traffic trends by source destination. This data can be used to help in network optimization for load-balancing traffic across alternate paths or by forwarding traffic to a preferred route.

Network monitoring—NetFlow data enables extensive near real-time network monitoring capabilities. You can use NetFlow flow data analysis to display traffic patterns associated with individual routing devices and switches as well as on a network-wide basis (providing aggregate traffic or application-based views) to provide proactive problem detection, efficient troubleshooting, and rapid problem resolution.

NetFlow Solutions Guide at § Why Deploy NetFlow?

1269. Further examples include NetFlow Solutions Guide at § NetFlow Feature Acceleration for Cisco IOS Features; NetFlow Configuration Guide at 3.

**3. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters to known thresholds.*

1270. Cisco NetFlow discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. For example:

NetFlow constitutes only one component of a comprehensive management data collection strategy. NetFlow is not intended as a replacement for SNMP and RMON/RMON2 capabilities. To maximize network monitoring, management, and planning capabilities, Cisco recommends you combine SNMP, RMON, and NetFlow data collection capabilities:

Use NetFlow with SNMP polling policies to gather key statistics on backbone or core routing devices and on MIB objects not related to flow-by-flow measurements including interface errors and memory and CPU usage statistics required for real-time monitoring.

Use NetFlow with RMON capabilities for detailed application tracking, interface error analysis and packet capture for problem diagnosis and resolution and for threshold-driven network management by way of events and alarms.

NetFlow Solutions Guide at § Using NetFlow with SNMP and RMON for the Ultimate Solution.

1271. Further examples include NetFlow Configuration Guide at 33–34.

1272. I also note Dijkstra’s self-stabilization algorithm was well-known at the time of the invention and would have been obvious to employ by the skilled artisan.

**4. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1273. Cisco NetFlow discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. For example: “Neighbour authentication can be configured for the following router protocols: . . . Open Shortest Path First . . .” ISP Essentials at 60–61.

1274. I also note Dijkstra’s self-stabilization algorithm was well-known at the time of the invention and would have been obvious to employ by the skilled artisan.

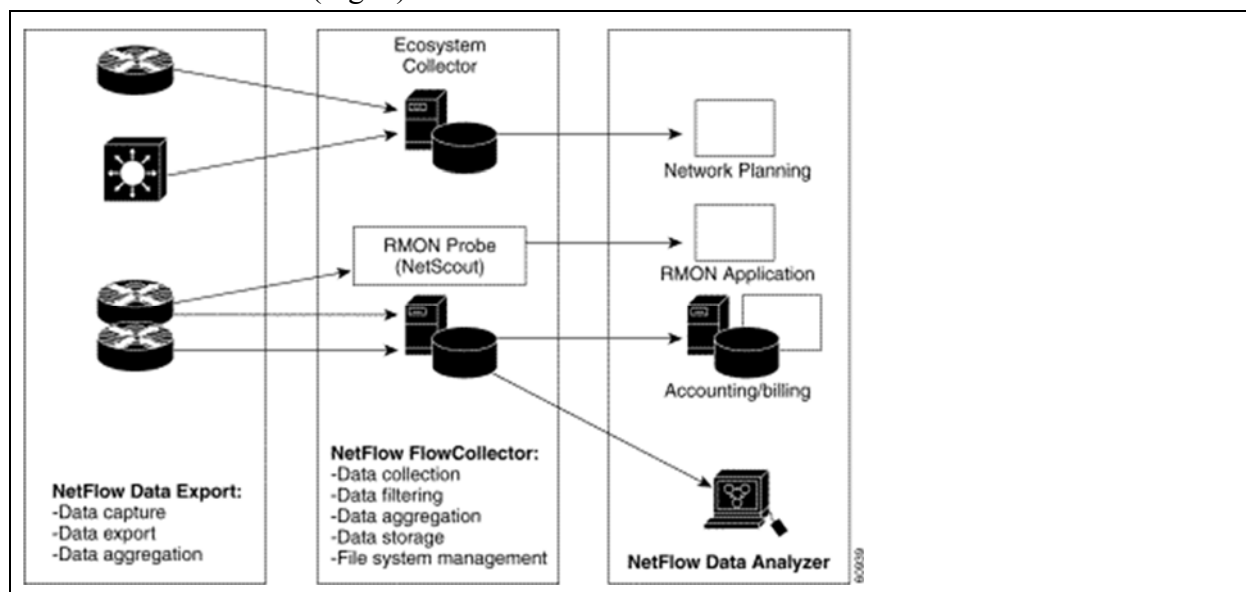
**5. Claim 7**

- a. *A non-transitory computer-readable medium comprising a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

1275. Cisco NetFlow discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . .” ’659 Pat. at Cl. 7. For example: “NetFlow Services infrastructure has a three-tiered architecture. Figure 1 illustrates three components: NetFlow Data Export on routing devices, the NFC, and the NDA.” NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1).



1276. Further examples include

b. *running at least one thread in a first runtime environment;*

1277. Cisco NetFlow discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1278. Cisco NetFlow discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

1279. Cisco NetFlow discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

- e. *performing a corrective action to fix any detected abnormalities;*

1280. Cisco NetFlow discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1281. Cisco NetFlow discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1282. Cisco NetFlow discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**6. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

1283. Cisco NetFlow discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2.

7. **Claim 9**

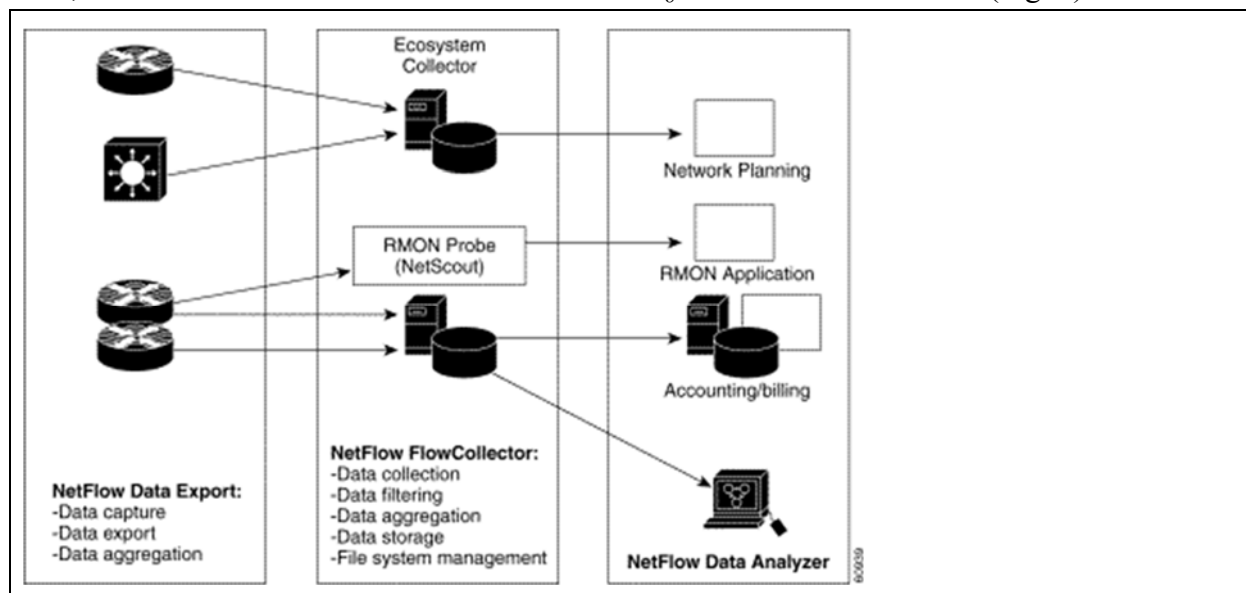
- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1284. Cisco NetFlow discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3.

8. **Claim 13**

- a. *A system, comprising:*

1285. Cisco NetFlow discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. For example: “NetFlow Services infrastructure has a three-tiered architecture. Figure 1 illustrates three components: NetFlow Data Export on routing devices, the NFC, and the NDA.” NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1).



1286. Further examples include NetFlow Configuration Guide at 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

1287. Cisco NetFlow discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. For example, “[t]his section provides information on CPU performance when NetFlow is enabled for a Route-Switch Processor2 (RSP2) with 128 MB of memory.” NetFlow Solutions Guide at § NetFlow CPU Usage Performance.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1288. Cisco NetFlow discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. For example, “[a]fter enabling NetFlow on an interface, NetFlow reserves memory allocated to accommodate a number of entries in the NetFlow cache.” NetFlow Solutions Guide at § Default NetFlow Cache Sizes.

d. *running at least one thread in a first runtime environment;*

1289. Cisco NetFlow discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1290. Cisco NetFlow discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c.

f. *detecting there is an abnormality in the monitored operational parameters;*

1291. Cisco NetFlow discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d.

g. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

1292. Cisco NetFlow discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1293. Cisco NetFlow discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1294. Cisco NetFlow discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1295. Cisco NetFlow discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

9. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*load balancing operation.*

1296. Cisco NetFlow discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2.

10. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1297. Cisco NetFlow discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3.

11. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1298. Cisco NetFlow discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

1299. I expect to testify that this art anticipates each Asserted Claim of the ’659 Patent. A claim chart illustrating that Turek discloses each and every limitation of those claims is included as Ex. B-1.

**IX. PRIOR ART COMBINATIONS RENDERING THE ’659 PATENT OBVIOUS**

**A. Turek in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek**

- 1. ***Motivation to Combine Turek with Buchanan<sup>6</sup>, Cantrill<sup>7</sup>, and/or***

---

<sup>6</sup> *Enhancing Network Management using Mobile Agents*, by WJ Buchanan, M. Naylor, and AV Scott, published January 10, 1997.

<sup>7</sup> *ThreadMon: A Tool for Monitoring Multithreaded Program Performance*, by Bryan M. Cantrill and Thomas W. Doeppner, Jr., published in 1997.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY*****Berry***<sup>8</sup>

1. Turek in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Turek discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Turek and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Turek because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial.. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Turek to include these elements, rendering them obvious.

---

<sup>8</sup> United States Patent No. 6,658,654, filed 6/6/2000 and issued to Berry, et al. on 12/2/2003.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. Motivation to Combine Turek with RFC 2328<sup>9</sup> and/or Afek<sup>10</sup>**

2. Turek in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Turek discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Turek and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Turek because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Turek to include these elements, rendering them obvious.

**3. Claim 1****a. A computer-implemented method, comprising:**

3. Turek discloses this claim element. *See* Turek Anticipation Claim 1.

**b. running at least one thread in a first runtime environment;**

4. Turek discloses this claim element. '659 Pat. at Cl. 1. *See* Turek Anticipation Claim

1.

---

<sup>9</sup> RFC 2328 "OSPF Version 2," published in April 1998 by the IETF Network Working Group.

<sup>10</sup> The local detection paradigm and its applications to self-stabilization, by Yehuda Afek, Shay Kutten, and Moti Yung, published in January 1996.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

5. To the extent Turek does not by itself anticipate this claim element, Turek in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Turek Anticipation Claim 1; Ex. B-1, claim 1.

6. Turek discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Turek Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

7. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry's network performance monitoring system:

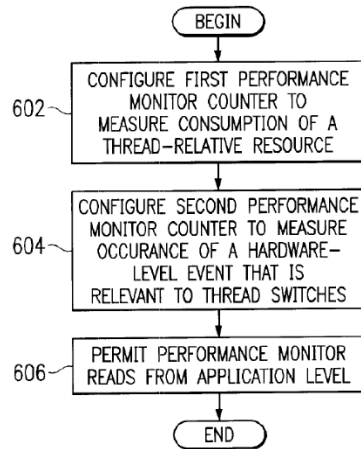
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Turek, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

8. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

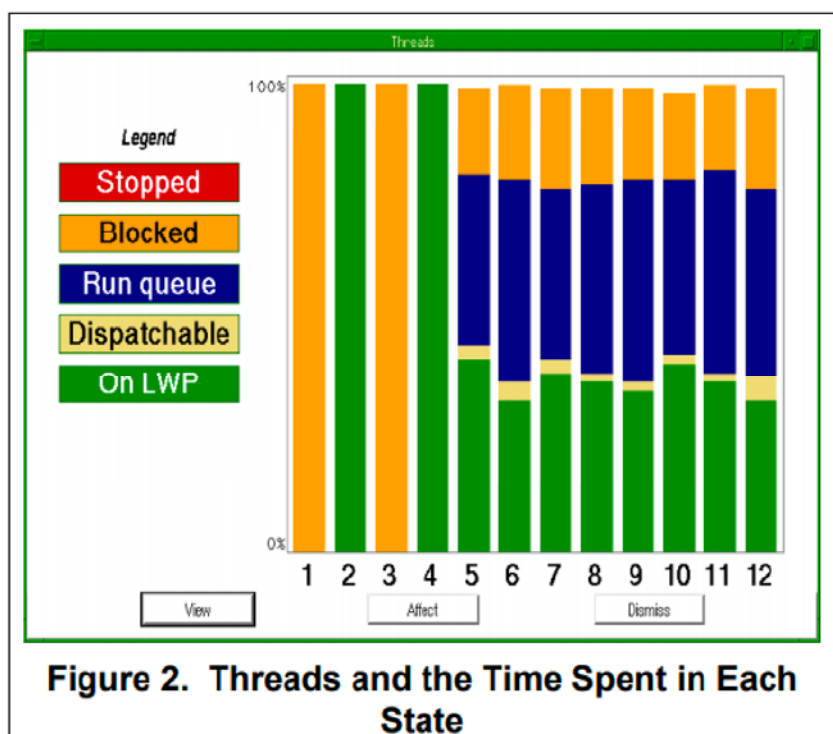
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors-any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult-it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Turek to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

d. *detecting if there is an abnormality in the monitored operational parameters;*

9. Turek discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Turek Anticipation Claim 1.

e. *and performing a corrective action to fix any detected abnormalities,*

10. Turek discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Turek Anticipation Claim 1.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

11. Turek discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Turek Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

12. Turek discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Turek Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

13. Turek discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. For example:

4. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

14. To the extent Turek does not by itself anticipate this claim element, Turek in combination with Cantrill and/or Berry discloses this claim element. ’659 Pat. at Cl. 1; *see* Turek Anticipation Claim 1; Turek Combination Claim 1, element c; Ex. B-1, claim 1.

15. Turek discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Turek Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

16. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Turek does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. **It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.*

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.*** The main advantages of threads are:

- o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as ***each thread can be tested independently of other threads.***
- o They can use standard threads, which are optimised for given hardware.

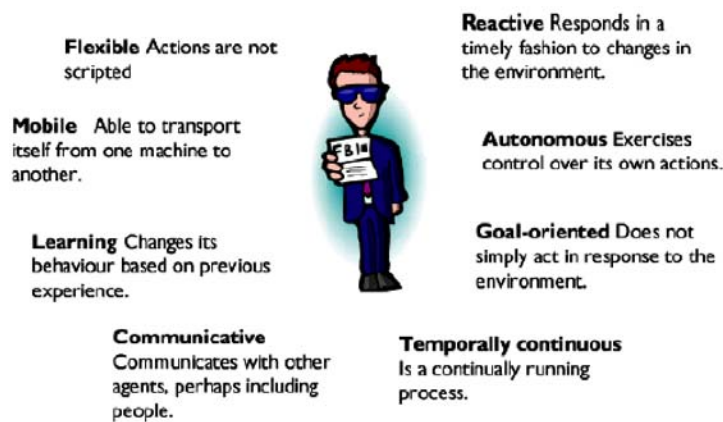
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

17. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

18. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Turek Combination Claim 1.

19. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Turek

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

5. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

20. Turek discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Turek Anticipation Claim 3.

6. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

21. Turek discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Turek Anticipation Claim 6.

22. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

23. To the extent Turek does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Turek in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Turek Anticipation Claims 1, 6; Turek Combination Claim 1; Ex. B-1, claim 6.

24. Turek discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Turek Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

25. To the extent this limitation is not expressly disclosed by Turek, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. See NetFuel's Infringement Contentions for '659 Patent. Turek discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

26. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Turek's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

27. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Turek to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Turek's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

7. **Claim 7**

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

28. Turek discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* Turek Anticipation Claim 7.

b. *running at least one thread in a first runtime environment;*

29. Turek discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Turek Anticipation Claim 7.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

30. Turek discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Turek Anticipation Claim 7. To the extent Turek does not by itself anticipate this claim element, Turek in combination with Cantrill and/or Berry discloses this claim element. *See* Turek Anticipation Claim 7; *see* Turek Combination Claim 1; Ex. B-1, claim 7.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

31. Turek discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Turek Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

32. Turek discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Turek Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

33. Turek discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Turek Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

34. Turek discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Turek Anticipation Claim 7.

8. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

35. Turek discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Turek Anticipation Claim 8. To the extent Turek does not by itself anticipate this claim element, Turek in combination with Buchanan and/or Cantrill discloses this claim element. *See* Turek Anticipation Claim 8; *see* Turek Combination Claim 2; Ex. B-1, claim 8.

9. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

36. Turek discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

known thresholds,” for at least the reasons explained above regarding claim 3. *See* Turek Anticipation Claim 9.

10. ***Claim 13***

a. *A system, comprising:*

37. Turek discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Turek Anticipation Claim 13.

b. *a processor;*

38. Turek discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Turek Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

39. Turek discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Turek Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

40. Turek discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Turek Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

41. Turek discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Turek Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

42. Turek discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Turek Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- g. *and performing a corrective action to fix any detected abnormalities;*

43. Turek discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Turek Anticipation Claim 13.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

44. Turek discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Turek Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

45. Turek discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Turek Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

46. Turek discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

47. Turek discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Turek Anticipation Claim 14. To the extent Turek does not by itself anticipate this claim element, Turek in combination with Buchanan and/or Cantrill discloses this claim element. *See* Turek Anticipation Claim 14; *see* Turek Combination Claim 2; Ex. B-1, claim 14.

12. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

48. Turek discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Turek Anticipation Claim 15.

13. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

49. Turek discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Turek does not by itself anticipate this claim element, Turek in combination with RFC 2328 and/or Afek discloses this claim element. *See* Turek Anticipation Claim 18; *see* Turek Combination Claim 6; Ex. B-1, claim 18.

**B. Goldman in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine Goldman with Buchanan, Cantrill, and/or Berry***

50. Goldman in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

management of distributed networks using software-agent-based scalable solutions. Goldman discloses using agents to assign and test policies on targets in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Goldman and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Goldman because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Goldman are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Goldman to include these elements, rendering them obvious.

2. ***Motivation to Combine Goldman with RFC 2328 and/or Afek***

51. Goldman in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Goldman discloses using agents to assign and test policies on targets in a distributed network; RFC 2328 discloses the Open

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Goldman and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Goldman because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Goldman are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Goldman to include these elements, rendering them obvious.

3. ***Motivation to Combine Goldman with Turek***

52. Goldman in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Goldman and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Goldman and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Goldman because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Goldman in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

multiple agents. Additionally, to the extent that Goldman and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Goldman and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

53. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

54. Goldman discloses this claim element. '659 Pat. at Cl. 1. *See* Goldman Anticipation Claim 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. B-3, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

55. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex B-3, claim 1.

56. Goldman discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Goldman Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

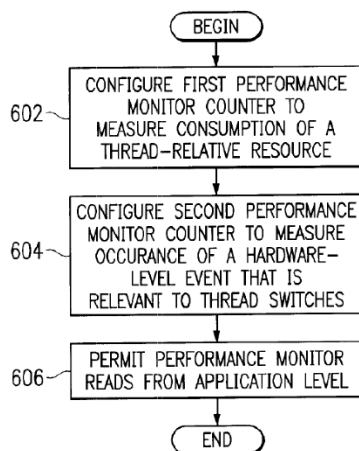
57. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



*FIG. 6A*

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Goldman, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

58. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

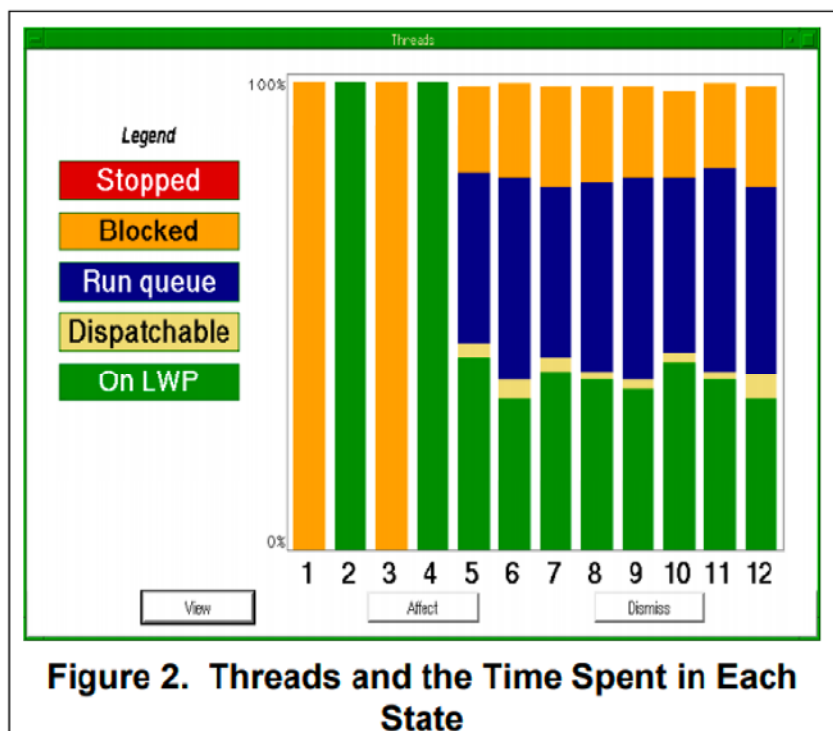
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Goldman to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

59. Goldman discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See Goldman Anticipation Claim 1.*

- e. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities,*

60. Goldman discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Goldman Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

61. Goldman discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Goldman Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

62. Goldman discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Goldman Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

63. Goldman discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Cantrill and/or Berry discloses this claim element. ’659 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. B-3, claim 1.

**5. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

64. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Goldman Anticipation Claim 2; Ex. B-3, claim 2.

65. Goldman discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Goldman Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan.

66. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Goldman does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

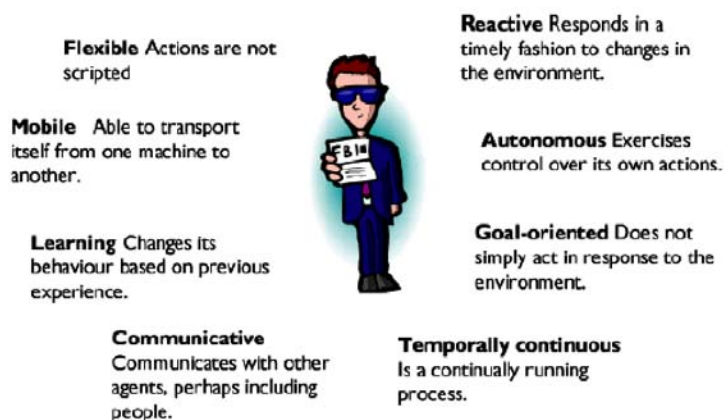
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

67. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*" Buchanan at sec. 4.

68. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Goldman to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan.

6. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

69. Goldman discloses claim 3, "[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds." '659 Pat. at Cl. 3. *See* Goldman Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

70. Goldman discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. *See* Goldman Anticipation Claim 6.

71. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

72. To the extent Goldman does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Goldman in combination with RFC 2328 and/or Afek discloses this claim element. '659 Pat. at Cl. 1; *see* Goldman Anticipation Claims 1, 6; Goldman Combination Claim 1; Ex. B-3, claim 6.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

73. Goldman discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Goldman Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

74. To the extent this limitation is not expressly disclosed by Goldman, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Goldman discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

75. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Goldman's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

76. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Goldman to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Goldman's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

77. Goldman discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Goldman Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

78. Goldman discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Goldman Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

79. Goldman discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Goldman Anticipation Claim 7. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Cantrill and/or Berry discloses this claim element. *See* Goldman Anticipation Claim 7; *see* Goldman Combination Claim 1; Ex. B-3, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

80. Goldman discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d. *See* Goldman Anticipation Claim 7.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

81. Goldman discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Goldman Anticipation Claim 7.*

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

82. Goldman discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See Goldman Anticipation Claim 7.*

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

83. Goldman discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See Goldman Anticipation Claim 7.*

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

84. Goldman discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See Goldman Anticipation Claim 8.* To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Buchanan and/or Cantrill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses this claim element. *See* Goldman Anticipation Claim 8; *see* Goldman Combination Claim 2; Ex. B-3, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

85. Goldman discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Goldman Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

86. Goldman discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Goldman Anticipation Claim 13.

- b. *a processor;*

87. Goldman discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Goldman Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

88. Goldman discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Goldman Anticipation Claim 13.

- d. *running at least one thread in a first runtime environment;*

89. Goldman discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Goldman Anticipation Claim 13.

- e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

90. Goldman discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Goldman Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

91. Goldman discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Goldman Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

92. Goldman discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Goldman Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

93. Goldman discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Goldman Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

94. Goldman discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Goldman Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

95. Goldman discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

96. Goldman discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Goldman Anticipation Claim 14. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Buchanan and/or Cantrill discloses this claim element. *See* Goldman Anticipation Claim 14; *see* Goldman Combination Claim 2; Ex. B-3, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

97. Goldman discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Goldman Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises using Dijkstra's Self Stabilization Algorithm.*

98. Goldman discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with RFC 2328 and/or Afek discloses this claim element. *See* Goldman Anticipation Claim 18; *see* Goldman Combination Claim 6; Ex. B-3, claim 18.

**C. Pandya in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek**

**1. *Motivation to Combine Pandya with Buchanan, Cantrill, and/or Berry***

99. Pandya in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Pandya discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Pandya and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill’s structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Pandya because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry’s systems enabled more operating parameters to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Pandya are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Pandya to include these elements, rendering them obvious.

2. ***Motivation to Combine Pandya with RFC 2328 and/or Afek***

100. Pandya in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Pandya discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Pandya and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Pandya because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Pandya are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Pandya to include these elements, rendering them obvious.

3. ***Claim 1***

a. *A computer-implemented method, comprising:*

101. Pandya discloses this claim element. *See Pandya Anticipation Claim 1.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

102. Pandya discloses this claim element. '659 Pat. at Cl. 1. *See Pandya Anticipation Claim 1.*

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

103. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see Pandya Anticipation Claim 1; Ex. B-4, claim 1.*

104. Pandya discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See Pandya Anticipation Claim 1.* It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

105. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry's network performance monitoring system:

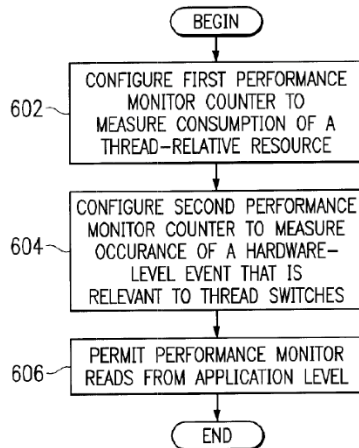


FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Pandya, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

***Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.***

In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. ***Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.*** Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

106. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

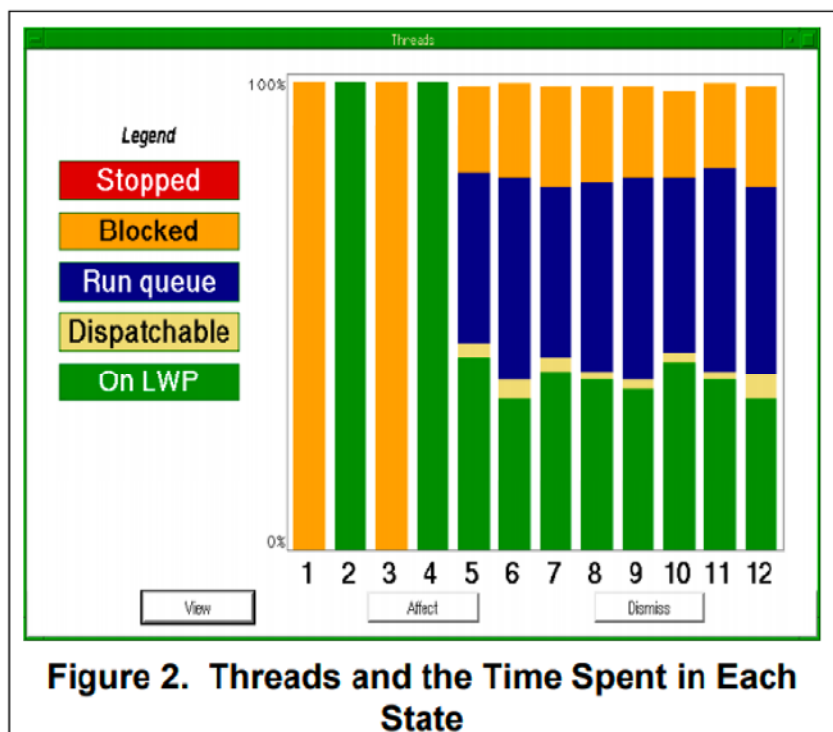
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler’s job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Pandya to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

107. Pandya discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Pandya Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

108. Pandya discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Pandya Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

109. Pandya discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Pandya Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

110. Pandya discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Pandya Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

111. Pandya discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1.

4. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

112. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-4, claim 2.

113. Pandya discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Pandya Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

114. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Pandya does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. **It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.*

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.*** The main advantages of threads are:

- o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as ***each thread can be tested independently of other threads.***
- o They can use standard threads, which are optimised for given hardware.

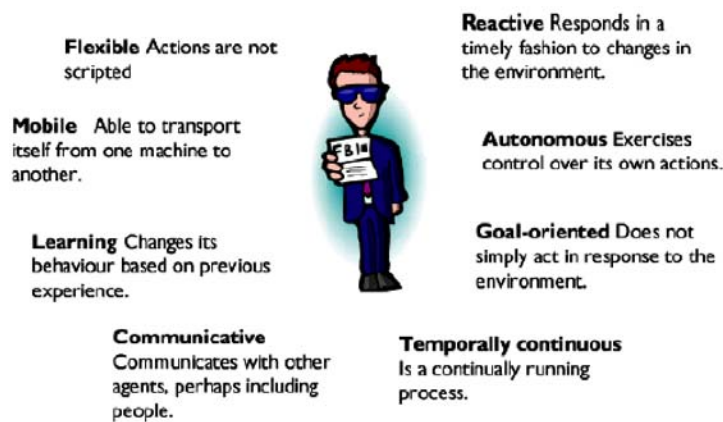
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

115. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

116. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Pandya Combination Claim 1.

117. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Pandya

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

**5. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

118. Pandya discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Pandya Anticipation Claim 3.

**6. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

119. Pandya discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Pandya Anticipation Claim 6.

120. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

121. To the extent Pandya does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Pandya in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Pandya Anticipation Claims 1, 6; Pandya Combination Claim 1; Ex. B-4, claim 6.

122. Pandya discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Pandya Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

123. To the extent this limitation is not expressly disclosed by Pandya, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. See NetFuel's Infringement Contentions for '659 Patent. Pandya discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

124. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Pandya's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

125. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Pandya to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Pandya's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

7. **Claim 7**

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

126. Pandya discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* Pandya Anticipation Claim 7.

b. *running at least one thread in a first runtime environment;*

127. Pandya discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Pandya Anticipation Claim 7.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

128. Pandya discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Pandya Anticipation Claim 7. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Cantrill and/or Berry discloses this claim element. *See* Pandya Anticipation Claim 7; *see* Pandya Combination Claim 1; Ex. B-4, claim 7.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

129. Pandya discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Pandya Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

130. Pandya discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Pandya Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

131. Pandya discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Pandya Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

132. Pandya discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Pandya Anticipation Claim 7.

8. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

133. Pandya discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Pandya Anticipation Claim 8. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Buchanan and/or Cantrill discloses this claim element. *See* Pandya Anticipation Claim 8; *see* Pandya Combination Claim 2; Ex. B-4, claim 8.

9. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

134. Pandya discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

known thresholds,” for at least the reasons explained above regarding claim 3. *See* Pandya Anticipation Claim 9.

10. ***Claim 13***

a. *A system, comprising:*

135. Pandya discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Pandya Anticipation Claim 13.

b. *a processor;*

136. Pandya discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Pandya Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

137. Pandya discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Pandya Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

138. Pandya discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Pandya Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

139. Pandya discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Pandya Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

140. Pandya discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Pandya Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- g. *and performing a corrective action to fix any detected abnormalities;*

141. Pandya discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Pandya Anticipation Claim 13.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

142. Pandya discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Pandya Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

143. Pandya discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Pandya Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

144. Pandya discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

145. Pandya discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Pandya Anticipation Claim 14. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Buchanan and/or Cantrill discloses this claim element. *See* Pandya Anticipation Claim 14; *see* Pandya Combination Claim 2; Ex. B-4, claim 14.

12. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

146. Pandya discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Pandya Anticipation Claim 15.

13. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

147. Pandya discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with RFC 2328 and/or Afek discloses this claim element. *See* Pandya Anticipation Claim 18; *see* Pandya Combination Claim 6; Ex. B-4, claim 18.

**D. Douik in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine Douik with Buchanan, Cantrill, and/or Berry***

148. Douik in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

management of distributed networks using software-agent-based scalable solutions. Douik discloses improving efficiency and scalability in a distributed network with agents having one or more associated underlying resources; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Douik and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Douik because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Douik are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Douik to include these elements, rendering them obvious.

2. ***Motivation to Combine Douik with RFC 2328 and/or Afek***

149. Douik in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Douik discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Douik and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Douik because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Douik are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Douik to include these elements, rendering them obvious.

3. ***Motivation to Combine Douik with Turek***

150. Douik in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Douik and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Douik and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Douik because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Douik in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agents. Additionally, to the extent that Douik and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Douik and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

151. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

152. Douik discloses this claim element. '659 Pat. at Cl. 1. *See* Douik Anticipation Claim 1. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Douik Anticipation Claim 1; Ex. B-5, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

153. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Douik Anticipation Claim 1; Ex. B-5, claim 1.

154. Douik discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Douik Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

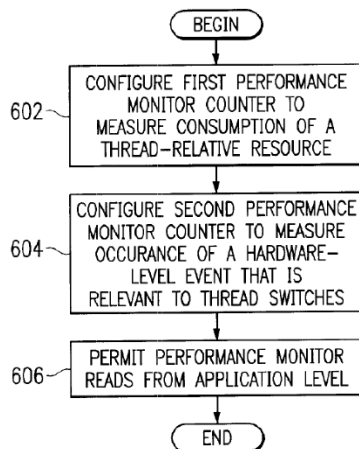
155. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



*FIG. 6A*

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Douik, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

156. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

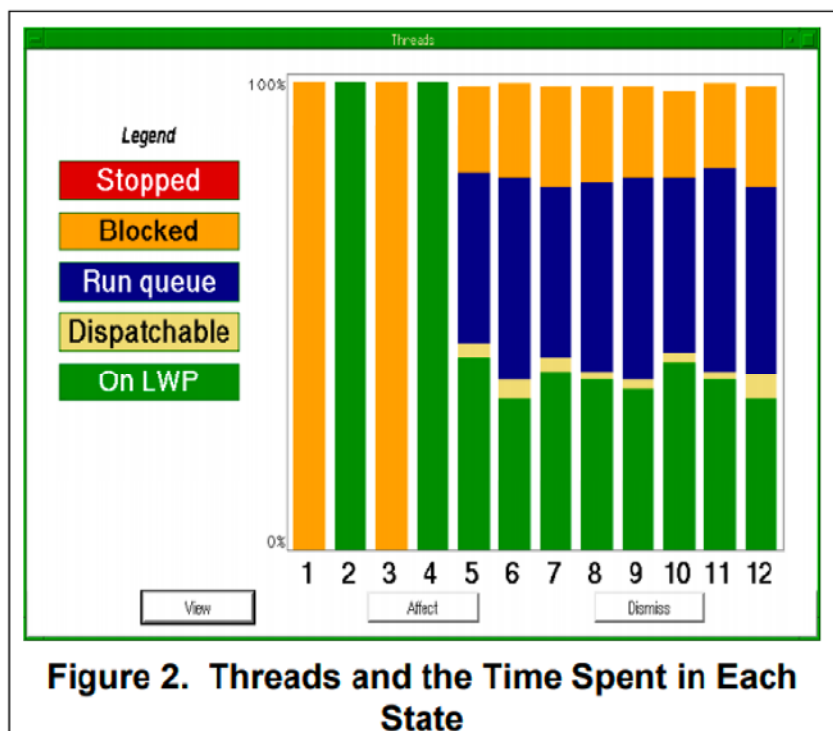
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Douik to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

157. Douik discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Douik Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities,*

158. Douik discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Douik Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

159. Douik discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Douik Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

160. Douik discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Douik Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

161. Douik discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Douik Anticipation Claim 1; Ex. B-5, claim 1; Turek Anticipation Claim 1.

**5. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

162. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-5, claim 2.

163. Douik discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Douik Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

164. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Douik does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

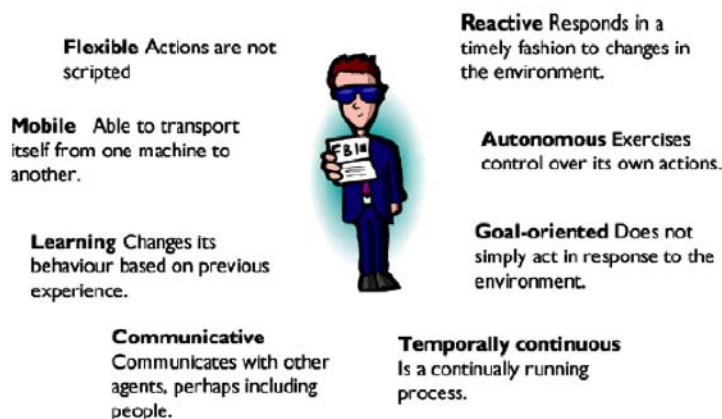
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

165. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

166. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Douik Combination Claim 1.

167. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Douik to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

168. Douik discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Douik Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

169. Douik discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Douik Anticipation Claim 6.

170. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

171. To the extent Douik does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Douik in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Douik Anticipation Claims 1, 6; Douik Combination Claim 1; Ex. B-5, claim 6.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

172. Douik discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Douik Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

173. To the extent this limitation is not expressly disclosed by Douik, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Douik discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

174. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Douik's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

175. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Douik



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Douik's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

176. Douik discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Douik Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

177. Douik discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Douik Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

178. Douik discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Douik Anticipation Claim 7. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Cantrill and/or Berry discloses this claim element. *See* Douik Anticipation Claim 7; *see* Douik Combination Claim 1; Ex. B-5, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

179. Douik discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d. *See* Douik Anticipation Claim 7.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

180. Douik discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Douik Anticipation Claim 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

181. Douik discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Douik Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

182. Douik discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Douik Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

183. Douik discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Douik Anticipation Claim 8. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Buchanan and/or Cantrill discloses this

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

claim element. *See* Douik Anticipation Claim 8; *see* Douik Combination Claim 2; Ex. B-5, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

184. Douik discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Douik Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

185. Douik discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Douik Anticipation Claim 13.

- b. *a processor;*

186. Douik discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Douik Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

187. Douik discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Douik Anticipation Claim 13.

- d. *running at least one thread in a first runtime environment;*

188. Douik discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Douik Anticipation Claim 13.

- e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

189. Douik discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Douik Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

190. Douik discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Douik Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

191. Douik discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Douik Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

192. Douik discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Douik Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

193. Douik discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Douik Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

194. Douik discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

195. Douik discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Douik Anticipation Claim 14. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Buchanan and/or Cantrill discloses this claim element. *See* Douik Anticipation Claim 14; *see* Douik Combination Claim 2; Ex. B-5, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

196. Douik discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Douik Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

197. Douik discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

explained above regarding claim 6. To the extent Douik does not by itself anticipate this claim element, Douik in combination with RFC 2328 and/or Afek discloses this claim element. *See* Douik Anticipation Claim 18; *see* Douik Combination Claim 6; Ex. B-5, claim 18.

**E. Natarajan in combination with Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

**1. *Motivation to Combine Natarajan with Cantrill, and/or Berry***

198. Natarajan in combination with Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Natarajan discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Natarajan and Cantrill and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Natarajan because the Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Cantrill and/or Berry and Natarajan are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

one of ordinary skill in the art, and it would have been obvious to modify Cantrill and/or Berry and/or Natarajan to include these elements, rendering them obvious.

**2. *Motivation to Combine Natarajan with RFC 2328 and/or Afek***

199. Natarajan in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Natarajan discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Natarajan and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Natarajan because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Natarajan are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Natarajan to include these elements, rendering them obvious.

**3. *Motivation to Combine Natarajan with Turek***

200. Natarajan in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Natarajan and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Natarajan and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Natarajan because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Natarajan in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Natarajan and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Natarajan and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

201. Natarajan discloses this claim element. *See* Natarajan Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

202. Natarajan discloses this claim element. '659 Pat. at Cl. 1. *See* Natarajan Anticipation Claim 1. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Natarajan Anticipation Claim 1; Ex. B-7, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

203. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Natarajan Anticipation Claim 1; Ex. B-7, claim 1.

204. Natarajan discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Natarajan Anticipation Claim

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

205. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



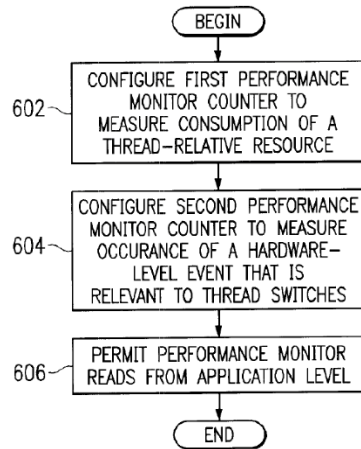
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Natarajan, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

206. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

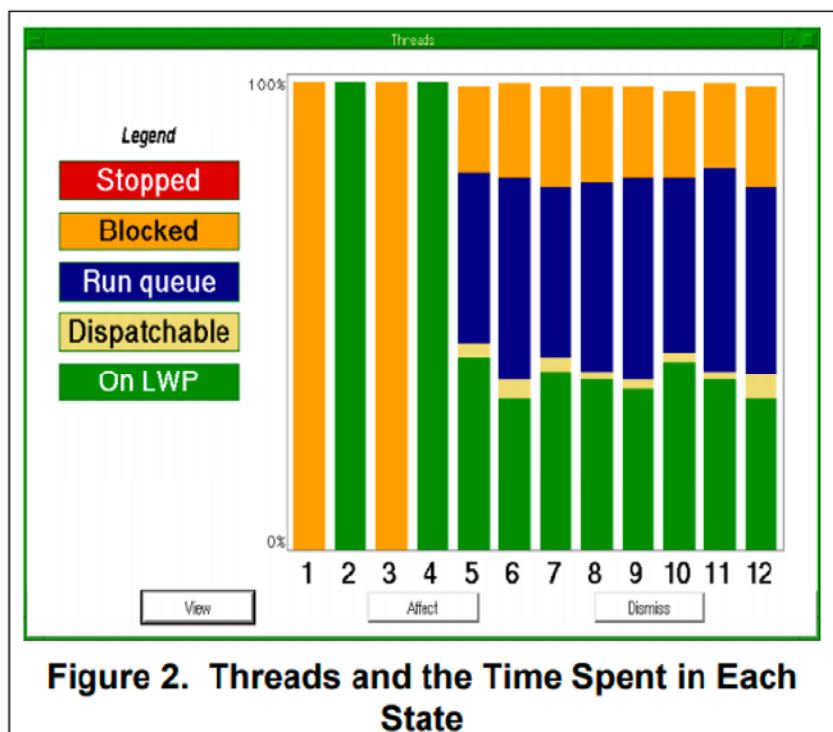
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Natarajan to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

207. Natarajan discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Natarajan Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

208. Natarajan discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Natarajan Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

209. Natarajan discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Natarajan Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

210. Natarajan discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Natarajan Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

211. Natarajan discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Natarajan Anticipation Claim 1; Ex. B-7, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

212. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Cantrill discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-7, claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

213. Natarajan discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Natarajan Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Cantrill.

214. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Natarajan Combination Claim 1.

215. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Natarajan to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Cantrill.

6. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

216. Natarajan discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Natarajan Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

217. Natarajan discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Natarajan Anticipation Claim 6.

218. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

219. To the extent Natarajan does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Natarajan in combination with RFC 2328 and/or Afek discloses this claim element. '659 Pat. at Cl. 1; *see* Natarajan Anticipation Claims 1, 6; Natarajan Combination Claim 1; Ex. B-7, claim 6.

220. Natarajan discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Natarajan Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

221. To the extent this limitation is not expressly disclosed by Natarajan, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Natarajan discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

222. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Natarajan's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

223. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Natarajan to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Natarajan's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

224. Natarajan discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Natarajan Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

225. Natarajan discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Natarajan Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

226. Natarajan discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Natarajan Anticipation Claim 7. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Cantrill and/or Berry discloses this claim element. *See* Natarajan Anticipation Claim 7; *see* Natarajan Combination Claim 1; Ex. B-7 claim 7.

- d. *detecting if an abnormality exists based on the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters;*

227. Natarajan discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Natarajan Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

228. Natarajan discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Natarajan Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

229. Natarajan discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Natarajan Anticipation Claim 7.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

230. Natarajan discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Natarajan Anticipation Claim 7.

9. ***Claim 8***

a. *The computer-readable medium of claim 7, wherein the corrective*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

231. Natarajan discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Natarajan Anticipation Claim 8. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Buchanan and/or Cantrill discloses this claim element. *See* Natarajan Anticipation Claim 8; *see* Natarajan Combination Claim 2; Ex. B-7, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

232. Natarajan discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Natarajan Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

233. Natarajan discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Natarajan Anticipation Claim 13.

- b. *a processor;*

234. Natarajan discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Natarajan Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

235. Natarajan discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Natarajan Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *running at least one thread in a first runtime environment;*

236. Natarajan discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Natarajan Anticipation Claim 13.*

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

237. Natarajan discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See Natarajan Anticipation Claim 13.*

f. *detecting there is an abnormality in the monitored operational parameters;*

238. Natarajan discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See Natarajan Anticipation Claim 13.*

g. *and performing a corrective action to fix any detected abnormalities;*

239. Natarajan discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Natarajan Anticipation Claim 13.*

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

240. Natarajan discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See Natarajan Anticipation Claim 13.*

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

241. Natarajan discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Natarajan Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

242. Natarajan discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

243. Natarajan discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Natarajan Anticipation Claim 14. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Buchanan and/or Cantrill discloses this claim element. *See* Natarajan Anticipation Claim 14; *see* Natarajan Combination Claim 2; Ex. B-7, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

244. Natarajan discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

thresholds,” at least for all the reasons explained above regarding claim 3. *See* Natarajan Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

245. Natarajan discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with RFC 2328 and/or Afek discloses this claim element. *See* Natarajan Anticipation Claim 18; *see* Natarajan Combination Claim 6; Ex. B-7, claim 18.

**F. Bonnell in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine Bonnell with Buchanan, Cantrill, and/or Berry***

246. Bonnell in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Bonnell discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Bonnell and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill’s structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Bonnell because the Buchanan, and/or Cantrill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Bonnell are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Bonnell to include these elements, rendering them obvious.

2. ***Motivation to Combine Bonnell with RFC 2328 and/or Afek***

247. Bonnell in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Bonnell discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Bonnell and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Bonnell because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Bonnell are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Bonnell to include these elements, rendering them obvious.

3. ***Motivation to Combine Bonnell with Turek***

248. Bonnell in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Bonnell and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Bonnell and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Bonnell because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Bonnell in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. As a further example, a person of ordinary skill in the art would have been motivated to implement Turek's policy requesting system in Bonnell because an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network, and because it is obvious to try with limited options. Additionally, to the extent that Bonnell and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Bonnell and/or Turek to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4. *Claim 1*

a. *A computer-implemented method, comprising:*

249. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

250. Bonnell discloses this claim element. '659 Pat. at Cl. 1. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. B-15, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

251. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. B-15, claim 1.

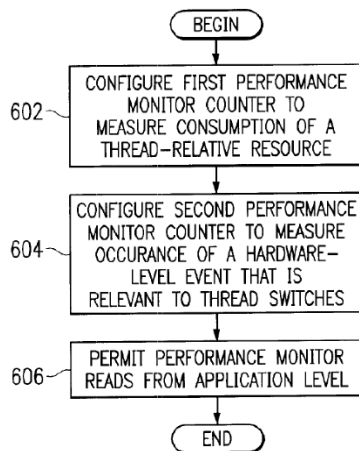
252. Bonnell discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Bonnell Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

253. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry's network performance monitoring system:



*FIG. 6A*

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Bonnell, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used. In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

254. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

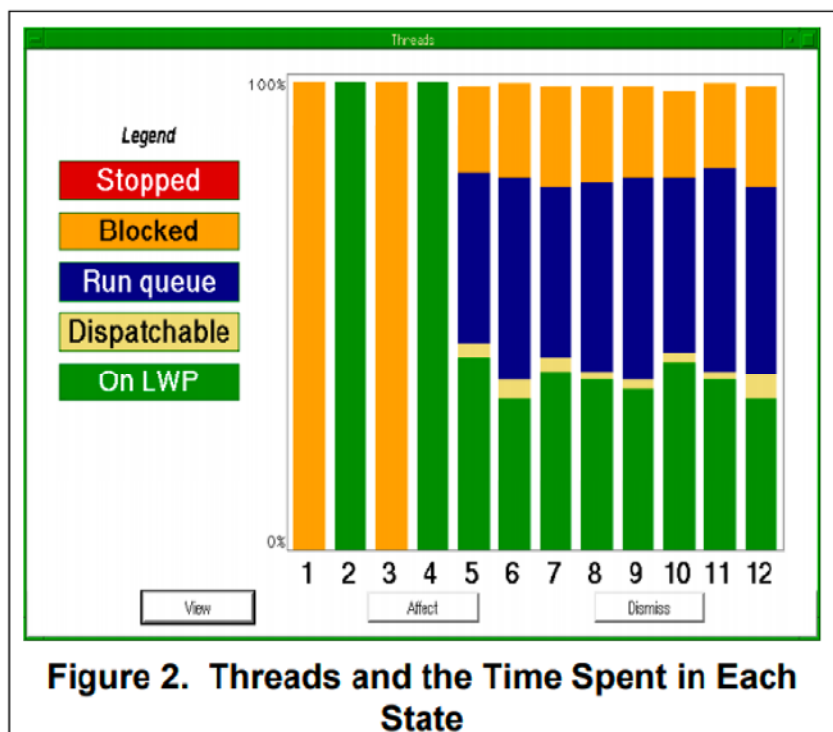
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Bonnell to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

255. Bonnell discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Bonnell Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

256. Bonnell discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Bonnell Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

257. Bonnell discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. B-15, claim 1; Turek Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

258. Bonnell discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Bonnell Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

259. Bonnell discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. B-15, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

260. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Bonnell Anticipation Claim 2; Ex. B-15, claim 2.

261. Bonnell discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Bonnell Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

262. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Bonnell does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple tasks simultaneously. Java makes creating, controlling, and coordinating threads relatively simple.* The main advantages of threads are:

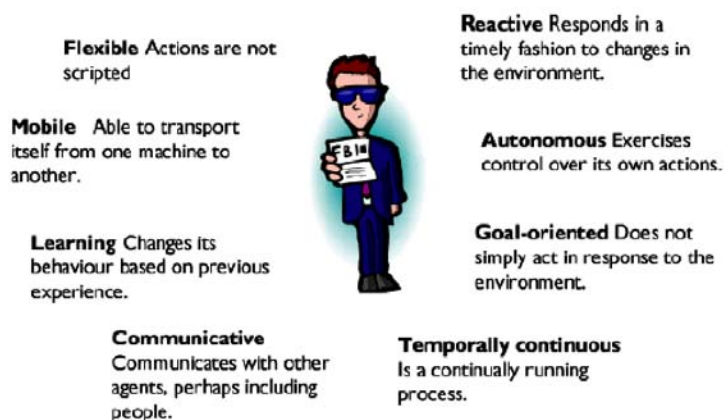
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

263. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*" Buchanan at sec. 4.

264. As described above, Cantrill also discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1; *see also* Bonnell Combination Claim 1.

265. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Bonnell to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

266. Bonnell discloses claim 3, "[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds." '659 Pat. at Cl. 3. *See* Bonnell Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

267. Bonnell discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. *See* Bonnell Anticipation Claim 6.

268. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

269. To the extent Bonnell does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Bonnell in combination

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

with RFC 2328 and/or Afek discloses this claim element. '659 Pat. at Cl. 1; *see* Bonnell Anticipation Claims 1, 6; Bonnell Combination Claim 1; Ex. B-15, claim 6.

270. Bonnell discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Bonnell Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

271. To the extent this limitation is not expressly disclosed by Bonnell, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Bonnell discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

272. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Bonnell's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

273. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Bonnell to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Bonnell's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

274. Bonnell discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Bonnell Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

275. Bonnell discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Bonnell Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

276. Bonnell discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Bonnell Anticipation Claim 7. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Cantrill and/or Berry discloses this claim element. *See* Bonnell Anticipation Claim 7; *see* Bonnell Combination Claim 1; Ex. B-15, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

277. Bonnell discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d. *See* Bonnell Anticipation Claim 7.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

278. Bonnell discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Bonnell Anticipation Claim 7.*

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

279. Bonnell discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See Bonnell Anticipation Claim 7.*

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

280. Bonnell discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See Bonnell Anticipation Claim 7.*

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

281. Bonnell discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See Bonnell Anticipation Claim 8.* To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Buchanan and/or Cantrill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses this claim element. *See* Bonnell Anticipation Claim 8; *see* Bonnell Combination Claim 2; Ex. B-15, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

282. Bonnell discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Bonnell Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

283. Bonnell discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Bonnell Anticipation Claim 13.

- b. *a processor;*

284. Bonnell discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Bonnell Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

285. Bonnell discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Bonnell Anticipation Claim 13.

- d. *running at least one thread in a first runtime environment;*

286. Bonnell discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Bonnell Anticipation Claim 13.

- e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

287. Bonnell discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Bonnell Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

288. Bonnell discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Bonnell Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

289. Bonnell discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Bonnell Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

290. Bonnell discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Bonnell Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

291. Bonnell discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Bonnell Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

292. Bonnell discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

293. Bonnell discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Bonnell Anticipation Claim 14. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Buchanan and/or Cantrill discloses this claim element. *See* Bonnell Anticipation Claim 14; *see* Bonnell Combination Claim 2; Ex. B-15, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

294. Bonnell discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Bonnell Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

295. Bonnell discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

reasons explained above regarding claim 6. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with RFC 2328 and/or Afek discloses this claim element. *See* Bonnell Anticipation Claim 18; *see* Bonnell Combination Claim 6; Ex. B-15, claim 18.

**G. Boukobza in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

**1. *Motivation to Combine Boukobza with Buchanan, Cantrill, and/or Berry***

296. Boukobza in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Boukobza discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boukobza and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Boukobza because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Boukobza are found to not include each

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Boukobza to include these elements, rendering them obvious.

**2. *Motivation to Combine Boukobza with RFC 2328 and/or Afek***

297. Boukobza in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Boukobza discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boukobza and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Boukobza because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Boukobza are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Boukobza to include these elements, rendering them obvious.

**3. *Motivation to Combine Boukobza with Turek***

298. Boukobza in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Boukobza and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boukobza and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Boukobza because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Boukobza in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Boukobza and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Boukobza and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

299. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

300. Boukobza discloses this claim element. '659 Pat. at Cl. 1. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. B-16, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

301. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. B-16, claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

302. Boukobza discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Boukobza Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

303. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



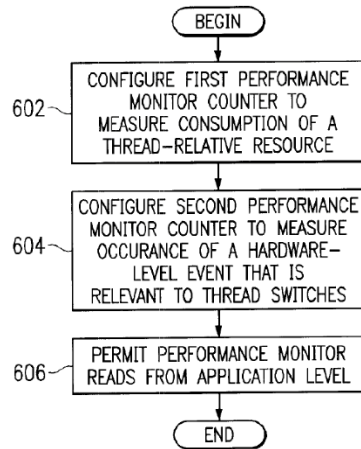
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Boukobza, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

304. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

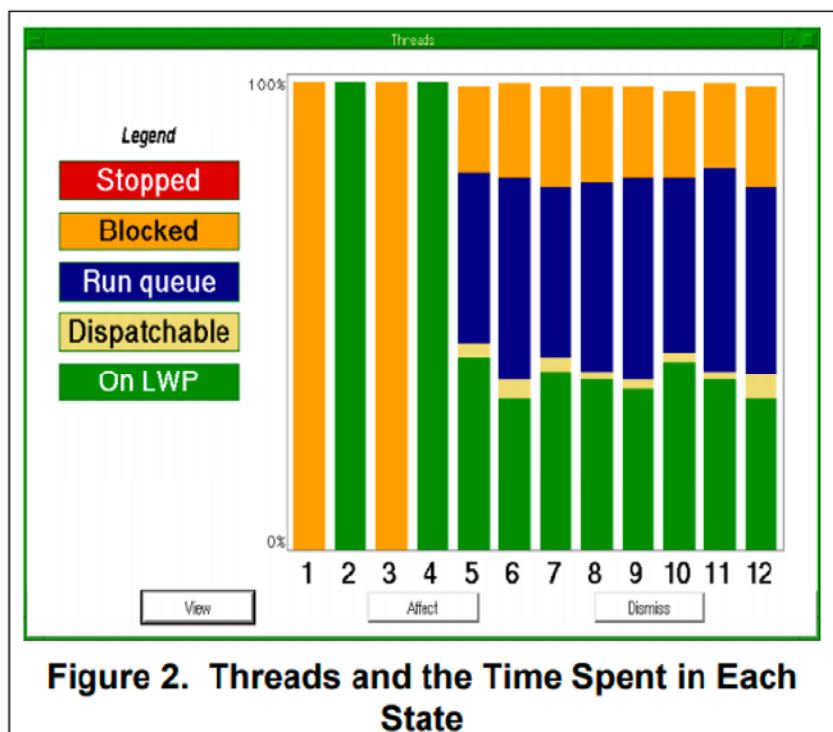
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boukobza to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

305. Boukobza discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Boukobza Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

306. Boukobza discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Boukobza Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

307. Boukobza discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Boukobza Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

308. Boukobza discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Boukobza Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

309. Boukobza discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. B-16, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

310. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Boukobza Anticipation Claim 2; Ex. B-16, claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

311. Boukobza discloses improving efficiency and scalability in a distributed network with agents having underlying resources. See Boukobza Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

312. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Boukobza does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. **It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.*

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.*** The main advantages of threads are:

o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

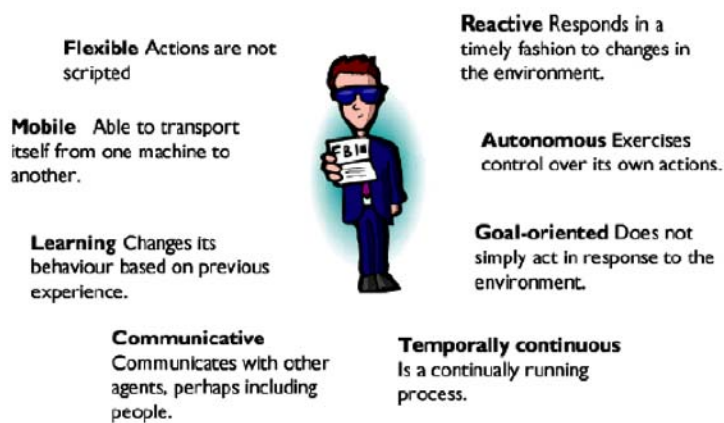
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

313. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

314. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Boukobza Combination Claim 1.

315. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boukobza to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

316. Boukobza discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Boukobza Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

317. Boukobza discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Boukobza Anticipation Claim 6.

318. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

319. To the extent Boukobza does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Boukobza in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Boukobza Anticipation Claims 1, 6; Boukobza Combination Claim 1; Ex. B-16, claim 6.

320. Boukobza discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Boukobza Anticipation Claims 1, 2. It would have



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

321. To the extent this limitation is not expressly disclosed by Boukobza, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Boukobza discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

322. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Boukobza's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

323. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boukobza to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Boukobza's



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

324. Boukobza discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* Boukobza Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

325. Boukobza discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Boukobza Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

326. Boukobza discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Boukobza Anticipation Claim 7. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Cantrill and/or Berry discloses this claim element. *See* Boukobza Anticipation Claim 7; *see* Boukobza Combination Claim 1; Ex. B-16, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

327. Boukobza discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Boukobza Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

328. Boukobza discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Boukobza Anticipation Claim 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

329. Boukobza discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Boukobza Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

330. Boukobza discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Boukobza Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

331. Boukobza discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Boukobza Anticipation Claim 8. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Buchanan and/or Cantrill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses this claim element. *See* Boukobza Anticipation Claim 8; *see* Boukobza Combination Claim 2; Ex. B-16, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

332. Boukobza discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Boukobza Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

333. Boukobza discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Boukobza Anticipation Claim 13.

- b. *a processor;*

334. Boukobza discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Boukobza Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

335. Boukobza discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Boukobza Anticipation Claim 13.

- d. *running at least one thread in a first runtime environment;*

336. Boukobza discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Boukobza Anticipation Claim 13.

- e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

337. Boukobza discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Boukobza Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

338. Boukobza discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Boukobza Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

339. Boukobza discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Boukobza Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

340. Boukobza discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Boukobza Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

341. Boukobza discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Boukobza Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

342. Boukobza discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

343. Boukobza discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Boukobza Anticipation Claim 14. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Buchanan and/or Cantrill discloses this claim element. *See* Boukobza Anticipation Claim 14; *see* Boukobza Combination Claim 2; Ex. B-16, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

344. Boukobza discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Boukobza Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises using Dijkstra's Self Stabilization Algorithm.*

345. Boukobza discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with RFC 2328 and/or Afek discloses this claim element. *See* Boukobza Anticipation Claim 18; *see* Boukobza Combination Claim 6; Ex. B-16, claim 18.

**H. Hellerstein in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

**1. *Motivation to Combine Hellerstein with Buchanan, Cantrill, and/or Berry***

346. Hellerstein in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Hellerstein discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Hellerstein and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill’s structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Hellerstein because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Hellerstein are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Hellerstein to include these elements, rendering them obvious.

2. ***Motivation to Combine Hellerstein with RFC 2328 and/or Afek***

347. Hellerstein in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Hellerstein discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Hellerstein and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Hellerstein because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Hellerstein are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Hellerstein to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****3. *Motivation to Combine Hellenstein with Turek***

348. Hellenstein in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Hellenstein and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Hellenstein and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Hellenstein because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Hellenstein in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. As a further example, a person of ordinary skill in the art would have been motivated to implement Turek's policy requesting system in Hellenstein because an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network, and because it is obvious to try with limited options. Additionally, to the extent that Hellenstein and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Hellenstein and/or Turek to include these elements, rendering them obvious.

**4. *Claim 1*****a. *A computer-implemented method, comprising:***

349. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

350. Hellerstein discloses this claim element. '659 Pat. at Cl. 1. *See* Hellerstein Anticipation Claim 1. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Hellerstein Anticipation Claim 1; Ex. B-20, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

351. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Hellerstein Anticipation Claim 1; Ex. B-20, claim 1.

352. Hellerstein discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Hellerstein Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

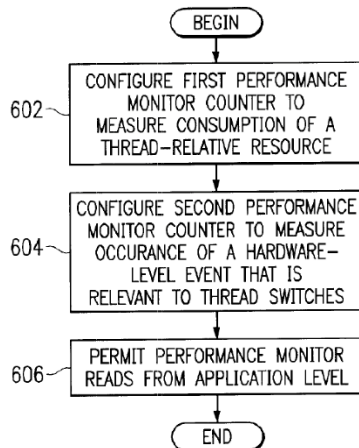
353. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. ***A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**602).** *A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry's network performance monitoring system:



**FIG. 6A**

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Hellerstein, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

***Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.*** In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. ***Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.*** Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

354. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

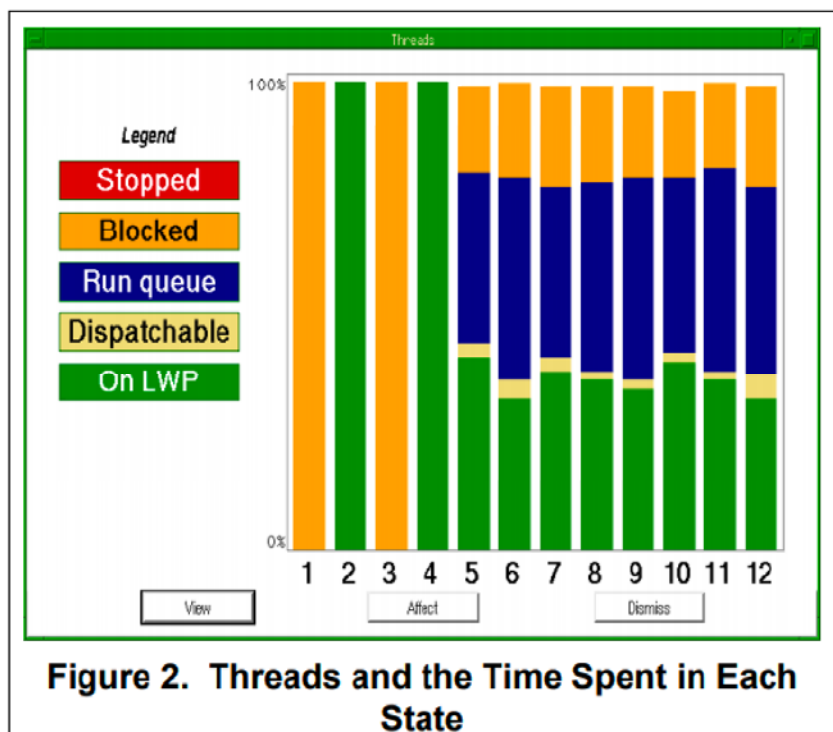
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Hellerstein to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

355. Hellerstein discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See Hellerstein Anticipation Claim 1.*

- e. *and performing a corrective action to fix any detected abnormalities,*

356. Hellerstein discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See Hellerstein Anticipation Claim 1.*

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

357. Hellerstein discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Hellerstein Anticipation Claim 1. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Hellerstein Anticipation Claim 1; Ex. B-20, claim 1; Turek Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

358. Hellerstein discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Hellerstein Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

359. Hellerstein discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Hellerstein Anticipation Claim 1; Ex. B-20, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

360. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Hellerstein Anticipation Claim 2; Ex. B-20, claim 2.

361. Hellerstein discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Hellerstein Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

362. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Hellerstein does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

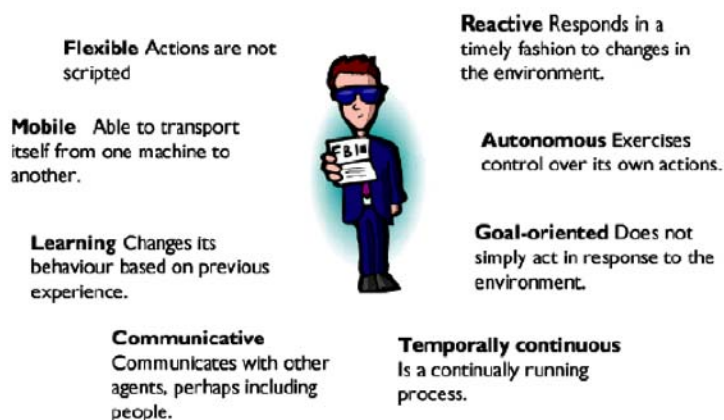
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

363. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*" Buchanan at sec. 4.

364. As described above, Cantrill also discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1; *see also* Hellerstein Combination Claim 1.

365. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Hellerstein to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

366. Hellerstein discloses claim 3, "[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds." '659 Pat. at Cl. 3. *See* Hellerstein Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

367. Hellerstein discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. *See* Hellerstein Anticipation Claim 6.

368. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

369. To the extent Hellerstein does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Hellerstein in



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

combination with RFC 2328 and/or Afek discloses this claim element. '659 Pat. at Cl. 1; *see* Hellerstein Anticipation Claims 1, 6; Hellerstein Combination Claim 1; Ex. B-20, claim 6.

370. Hellerstein discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Hellerstein Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

371. To the extent this limitation is not expressly disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Hellerstein discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

372. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Hellerstein's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

373. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Hellerstein to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Hellerstein's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

374. Hellerstein discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Hellerstein Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

375. Hellerstein discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Hellerstein Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

376. Hellerstein discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Hellerstein Anticipation Claim 7. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Cantrill and/or Berry discloses this claim element. *See* Hellerstein Anticipation Claim 7; *see* Hellerstein Combination Claim 1; Ex. B-20, claim 7.

- d. *detecting if an abnormality exists based on the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters;*

377. Hellerstein discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Hellerstein Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

378. Hellerstein discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Hellerstein Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

379. Hellerstein discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Hellerstein Anticipation Claim 7.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

380. Hellerstein discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Hellerstein Anticipation Claim 7.

9. ***Claim 8***

a. *The computer-readable medium of claim 7, wherein the corrective*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

381. Hellerstein discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Hellerstein Anticipation Claim 8. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Buchanan and/or Cantrill discloses this claim element. *See* Hellerstein Anticipation Claim 8; *see* Hellerstein Combination Claim 2; Ex. B-20, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

382. Hellerstein discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Hellerstein Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

383. Hellerstein discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Hellerstein Anticipation Claim 13.

- b. *a processor;*

384. Hellerstein discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Hellerstein Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

385. Hellerstein discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Hellerstein Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *running at least one thread in a first runtime environment;*

386. Hellerstein discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Hellerstein Anticipation Claim 13.*

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

387. Hellerstein discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See Hellerstein Anticipation Claim 13.*

f. *detecting there is an abnormality in the monitored operational parameters;*

388. Hellerstein discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See Hellerstein Anticipation Claim 13.*

g. *and performing a corrective action to fix any detected abnormalities;*

389. Hellerstein discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Hellerstein Anticipation Claim 13.*

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

390. Hellerstein discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See Hellerstein Anticipation Claim 13.*

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

391. Hellerstein discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Hellerstein Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

392. Hellerstein discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

393. Hellerstein discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Hellerstein Anticipation Claim 14. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Buchanan and/or Cantrill discloses this claim element. *See* Hellerstein Anticipation Claim 14; *see* Hellerstein Combination Claim 2; Ex. B-20, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

394. Hellerstein discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

thresholds,” at least for all the reasons explained above regarding claim 3. *See* Hellerstein Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

395. Hellerstein discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with RFC 2328 and/or Afek discloses this claim element. *See* Hellerstein Anticipation Claim 18; *see* Hellerstein Combination Claim 6; Ex. B-20, claim 18.

**I. Boudaoud in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine Boudaoud with Buchanan, Cantrill, and/or Berry***

396. Boudaoud in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Boudaoud discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boudaoud and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill’s structure for monitoring numerous operating parameters

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

of a single core in the multicore system disclosed in Boudaoud because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Boudaoud are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Boudaoud to include these elements, rendering them obvious.

**2. *Motivation to Combine Boudaoud with RFC 2328 and/or Afek***

397. Boudaoud in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Boudaoud discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boudaoud and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Boudaoud because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Boudaoud are found to not include each element of the asserted claims of the '659 Patent, these elements were



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Boudaoud to include these elements, rendering them obvious.

3. ***Motivation to Combine Boudaoud with Turek***

398. Boudaoud in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Boudaoud and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boudaoud and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Boudaoud because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Boudaoud in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Boudaoud and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Boudaoud and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

399. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

400. Boudaoud discloses this claim element. '659 Pat. at Cl. 1. *See* Boudaoud Anticipation Claim 1. To the extent Boudaoud does not by itself anticipate this claim element,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Boudaoud in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Boudaoud Anticipation Claim 1; Ex. B-14, claim 1; Turek Anticipation Claim 1.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

401. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Boudaoud Anticipation Claim 1; Ex. B-14, claim 1.

402. Boudaoud discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Boudaoud Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

403. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

*With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry's network performance monitoring system:

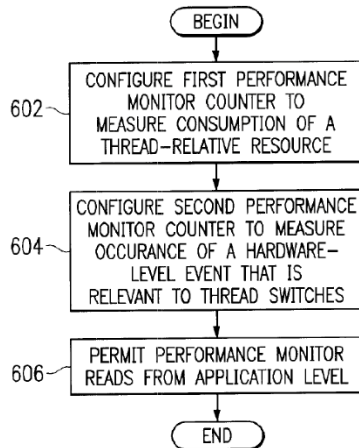


FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Boudaoud, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.*

In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

404. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

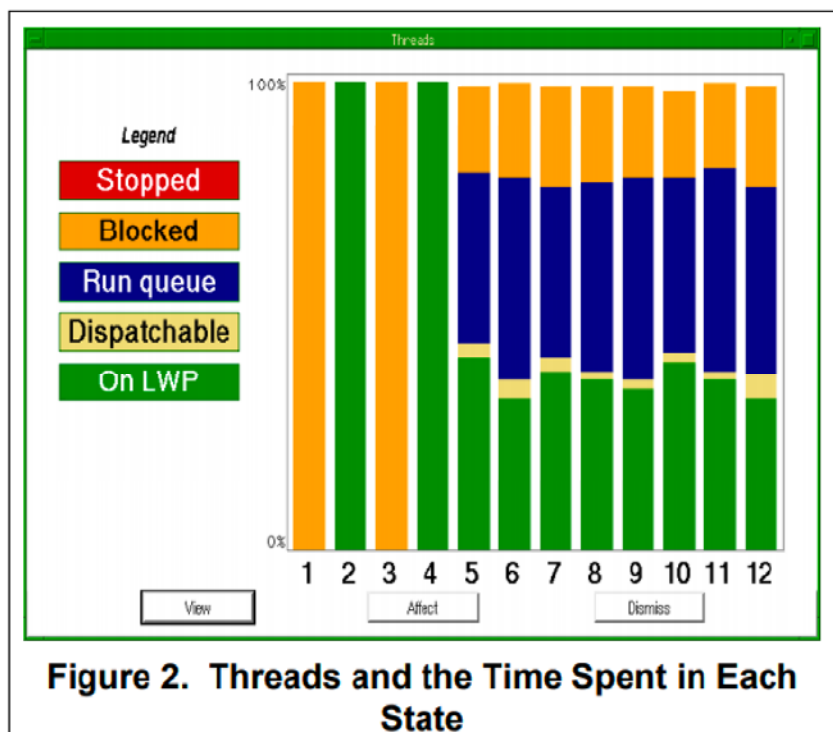
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler’s job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boudaoud to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

405. Boudaoud discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Boudaoud Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

406. Boudaoud discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Boudaoud Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

407. Boudaoud discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Boudaoud Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

408. Boudaoud discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Boudaoud Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

409. Boudaoud discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Boudaoud Anticipation Claim 1; Ex. B-14, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

410. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-14, claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

411. Boudaoud discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Boudaoud Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

412. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Boudaoud does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. **It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.*

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.*** The main advantages of threads are:

o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

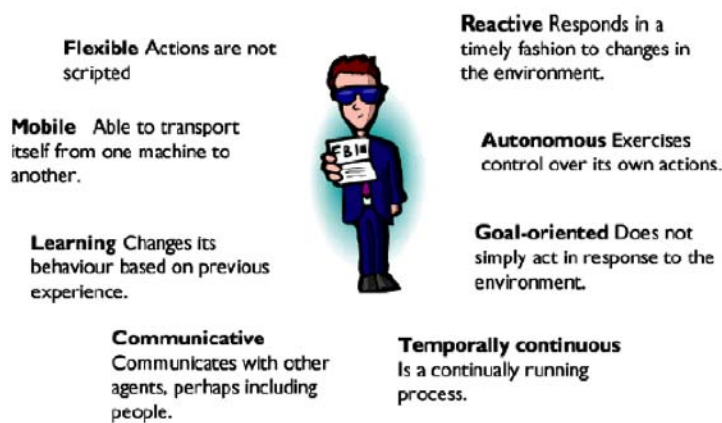
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

413. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

414. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Boudaoud Combination Claim 1.

415. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boudaoud to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

416. Boudaoud discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Boudaoud Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

417. Boudaoud discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Boudaoud Anticipation Claim 6.

418. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

419. To the extent Boudaoud does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Boudaoud in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Boudaoud Anticipation Claims 1, 6; Boudaoud Combination Claim 1; Ex. B-14, claim 6.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

420. Boudaoud discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Boudaoud Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

421. To the extent this limitation is not expressly disclosed by Boudaoud, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Boudaoud discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

422. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Boudaoud's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

423. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Boudaoud to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Boudaoud's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

424. Boudaoud discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Boudaoud Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

425. Boudaoud discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Boudaoud Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

426. Boudaoud discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Boudaoud Anticipation Claim 7. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Cantrill and/or Berry discloses this claim element. *See* Boudaoud Anticipation Claim 7; *see* Boudaoud Combination Claim 1; Ex. B-14, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

427. Boudaoud discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d. *See* Boudaoud Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

428. Boudaoud discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Boudaoud Anticipation Claim 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

429. Boudaoud discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Boudaoud Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

430. Boudaoud discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Boudaoud Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

431. Boudaoud discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Boudaoud Anticipation Claim 8. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Buchanan and/or Cantrill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses this claim element. *See* Boudaoud Anticipation Claim 8; *see* Boudaoud Combination Claim 2; Ex. B-14, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

432. Boudaoud discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Boudaoud Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

433. Boudaoud discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Boudaoud Anticipation Claim 13.

- b. *a processor;*

434. Boudaoud discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Boudaoud Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

435. Boudaoud discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Boudaoud Anticipation Claim 13.

- d. *running at least one thread in a first runtime environment;*

436. Boudaoud discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Boudaoud Anticipation Claim 13.

- e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

437. Boudaoud discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Boudaoud Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

438. Boudaoud discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Boudaoud Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

439. Boudaoud discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Boudaoud Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

440. Boudaoud discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Boudaoud Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

441. Boudaoud discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Boudaoud Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

442. Boudaoud discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

443. Boudaoud discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Boudaoud Anticipation Claim 14. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Buchanan and/or Cantrill discloses this claim element. *See* Boudaoud Anticipation Claim 14; *see* Boudaoud Combination Claim 2; Ex. B-14, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

444. Boudaoud discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Boudaoud Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises using Dijkstra's Self Stabilization Algorithm.*

445. Boudaoud discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with RFC 2328 and/or Afek discloses this claim element. *See* Boudaoud Anticipation Claim 18; *see* Boudaoud Combination Claim 6; Ex. B-14, claim 18.

**J. Kasteleijn in combination with Cantrill/Berry and/or RFC 2328 and/or Lindskog**

**1. *Motivation to Combine Kasteleijn with Cantrill, and/or Berry***

446. Kasteleijn in combination with Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Kasteleijn discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Kasteleijn and Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Cantrill’s structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Kasteleijn because the Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry’s systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

beneficial. Additionally, to the extent that Cantrill, and/or Berry and Kasteleijn are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Cantrill, and/or Berry and/or Kasteleijn to include these elements, rendering them obvious.

2. ***Motivation to Combine Kasteleijn with RFC 2328***

447. Kasteleijn in combination with RFC 2328 renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Kasteleijn discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Kasteleijn and RFC 2328. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328's self-stabilizing algorithms in Kasteleijn because the RFC 2328 techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and Kasteleijn are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Kasteleijn to include these elements, rendering them obvious.

3. ***Motivation to Combine Kasteleijn with Lindskog***

448. Kasteleijn in combination with Lindskog renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Kasteleijn and Lindskog disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Kasteleijn and Lindskog. For example, a person of ordinary skill in the art would have been motivated to implement Lindskog's corrective actions in Kasteleijn because the MCs of Kasteleijn can be programmed to perform any number of management tasks, and a POSITA would have been motivated to achieve the predictable result of including correcting detected abnormalities as disclosed by Lindskog as one of the management tasks, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Kasteleijn and Lindskog are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Kasteleijn and/or Lindskog to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

449. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

450. Kasteleijn discloses this claim element. '659 Pat. at Cl. 1. *See* Kasteleijn Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

451. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. B-2, claim 1.

452. Kasteleijn discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Kasteleijn Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

453. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

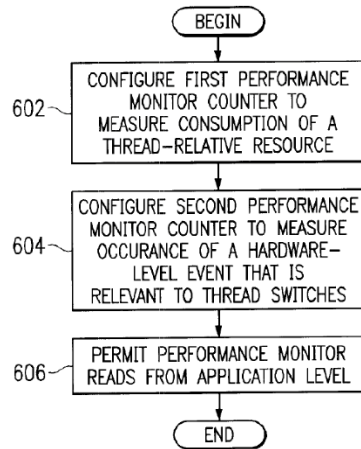
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Kasteleijn, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

454. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

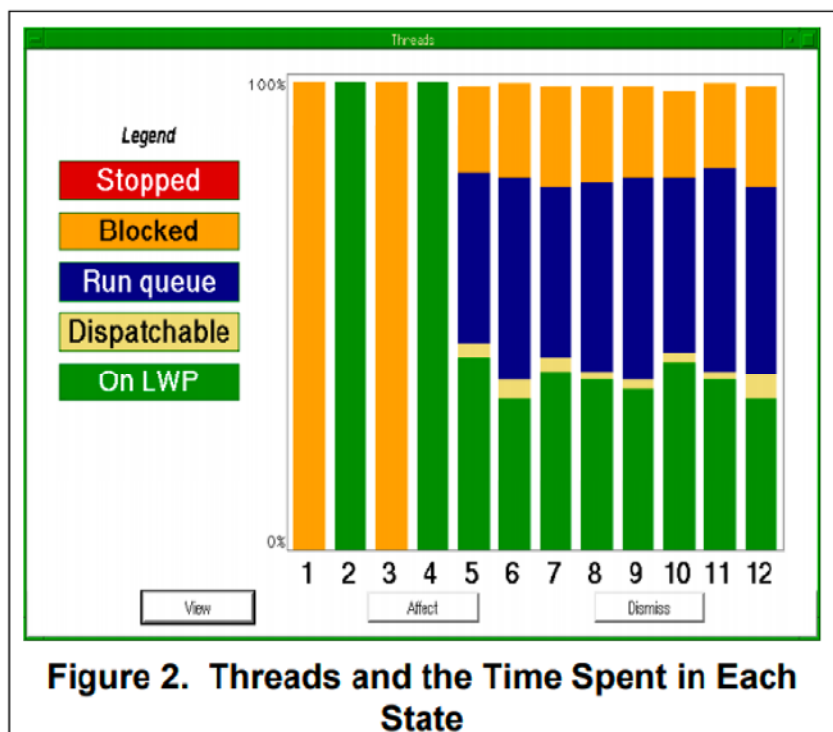
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Kasteleijn to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

455. Kasteleijn discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Kasteleijn Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

456. Kasteleijn discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Kasteleijn Anticipation Claim 1. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Lindskog discloses this claim element. ’659 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. B-2, claim 1; Lindskog Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

457. Kasteleijn discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Kasteleijn Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

458. Kasteleijn discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Kasteleijn Anticipation Claim 1. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Lindskog discloses this claim element. ’659 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. B-2, claim 1; Lindskog Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

459. Kasteleijn discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1.

5. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

460. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-2, claim 2.

461. Kasteleijn discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Kasteleijn Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Cantrill.

462. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” '659 Pat., Cl. 1; *see also* Kasteleijn Combination Claim 1.

463. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Kasteleijn to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

464. Kasteleijn discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” '659 Pat. at Cl. 3. *See* Kasteleijn Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

465. Kasteleijn discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.” '659 Pat. at Cl. 6. *See* Kasteleijn Anticipation Claim 6.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

466. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

467. To the extent Kasteleijn does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Kasteleijn in combination with RFC 2328 discloses this claim element. '659 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claims 1, 6; Kasteleijn Combination Claim 1; Ex. B-2, claim 6.

468. Kasteleijn discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Kasteleijn Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 .

469. To the extent this limitation is not expressly disclosed by Kasteleijn, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Kasteleijn discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

470. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Kasteleijn to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Kasteleijn's multi-threaded distributed networking management system, as disclosed by RFC 2328 .

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

471. Kasteleijn discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* Kasteleijn Anticipation Claim 7.

b. *running at least one thread in a first runtime environment;*

472. Kasteleijn discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Kasteleijn Anticipation Claim 7.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

473. Kasteleijn discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Kasteleijn Anticipation Claim 7. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Cantrill and/or Berry discloses this claim element. *See* Kasteleijn Anticipation Claim 7; *see* Kasteleijn Combination Claim 1; Ex. B-2, claim 7.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

474. Kasteleijn discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Kasteleijn Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

475. Kasteleijn discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Kasteleijn Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

476. Kasteleijn discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Kasteleijn Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

477. Kasteleijn discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Kasteleijn Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

478. Kasteleijn discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Kasteleijn Anticipation Claim 8. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Cantrill discloses this claim element. *See* Kasteleijn Anticipation Claim 8; *see* Kasteleijn Combination Claim 2; Ex. B-2, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

479. Kasteleijn discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

known thresholds,” for at least the reasons explained above regarding claim 3. *See* Kasteleijn Anticipation Claim 9.

11. ***Claim 13***

a. *A system, comprising:*

480. Kasteleijn discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Kasteleijn Anticipation Claim 13.

b. *a processor;*

481. Kasteleijn discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Kasteleijn Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

482. Kasteleijn discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Kasteleijn Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

483. Kasteleijn discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Kasteleijn Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

484. Kasteleijn discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Kasteleijn Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

485. Kasteleijn discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Kasteleijn Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

486. Kasteleijn discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Kasteleijn Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

487. Kasteleijn discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Kasteleijn Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

488. Kasteleijn discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Kasteleijn Anticipation Claim 13.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

489. Kasteleijn discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

490. Kasteleijn discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Kasteleijn Anticipation Claim 14. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Cantrill discloses this claim element. *See* Kasteleijn Anticipation Claim 14; *see* Kasteleijn Combination Claim 2; Ex. B-2, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

491. Kasteleijn discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Kasteleijn Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

492. Kasteleijn discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with RFC 2328 discloses this claim element. *See* Kasteleijn Anticipation Claim 18; *see* Kasteleijn Combination Claim 6; Ex. B-2, claim 18.

**K. Da Rocha in combination with Buchanan/Cantrill/Berry and/or RFC**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2328/Afek and/or Turek****1. *Motivation to Combine Da Rocha with Buchanan, Cantrill, and/or Berry***

493. Da Rocha in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Da Rocha discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Da Rocha and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Da Rocha because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Da Rocha are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Da Rocha to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. Motivation to Combine Da Rocha with RFC 2328 and/or Afek**

494. Da Rocha in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Da Rocha discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Da Rocha and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Da Rocha because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Da Rocha are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Da Rocha to include these elements, rendering them obvious.

**3. Motivation to Combine da Rocha with Turek**

495. da Rocha in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both da Rocha and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of da Rocha and Turek. For example, a person of ordinary skill



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

in the art would have been motivated to implement Turek's multithreading in da Rocha because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in da Rocha in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that da Rocha and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify da Rocha and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

496. Da Rocha discloses this claim element. *See* Da Rocha Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

497. Da Rocha discloses this claim element. '659 Pat. at Cl. 1. *See* Da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. B-6, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

498. To the extent Da Rocha does not by itself anticipate this claim element, Da Rocha in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Da Rocha Anticipation Claim 1; Ex. B-6, claim 1.

499. Da Rocha discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Da Rocha Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

500. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

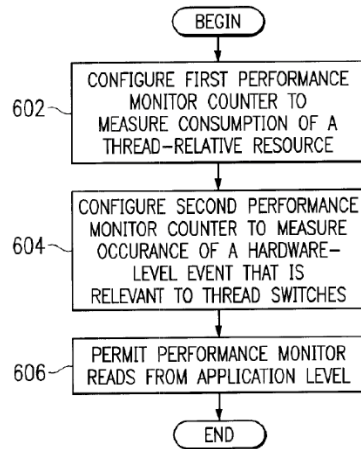
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Da Rocha, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

501. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

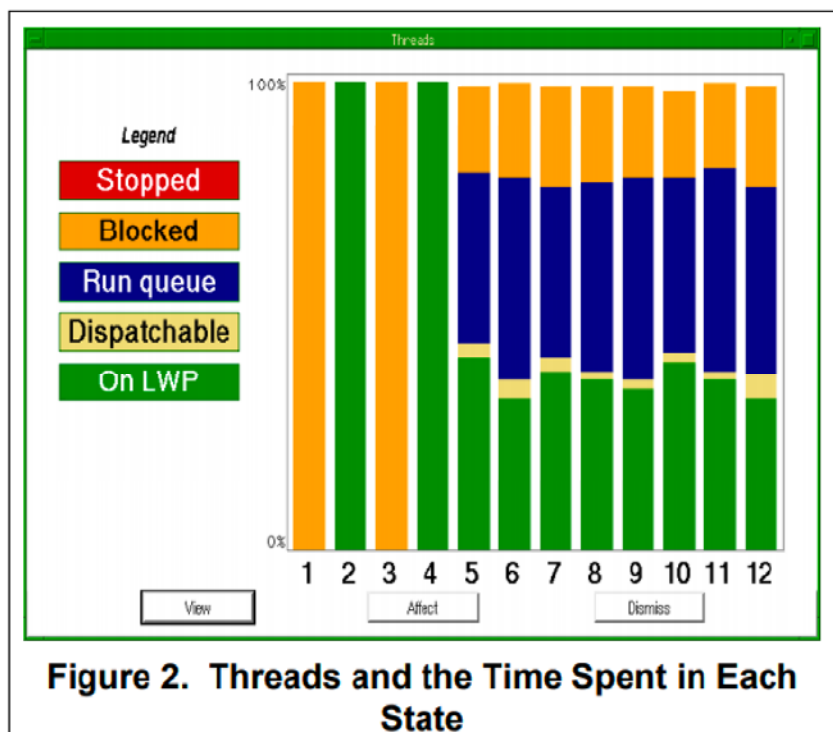
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Da Rocha to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

502. Da Rocha discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Da Rocha Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

503. Da Rocha discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Da Rocha Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

504. Da Rocha discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Da Rocha Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

505. Da Rocha discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Da Rocha Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

506. Da Rocha discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. B-6, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

507. To the extent Da Rocha does not by itself anticipate this claim element, Da Rocha in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-6, claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

508. Da Rocha discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Da Rocha Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

509. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Da Rocha does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. **It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.*

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.*** The main advantages of threads are:

o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

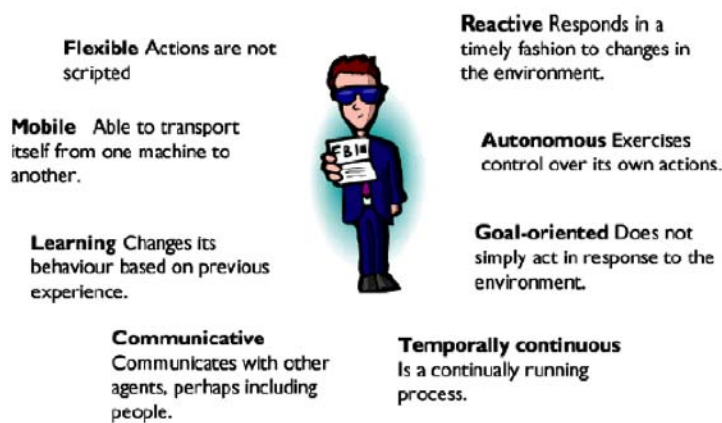
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

510. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

511. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Da Rocha Combination Claim 1.

512. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Da Rocha to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

513. Da Rocha discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Da Rocha Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

514. Da Rocha discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Da Rocha Anticipation Claim 6.

515. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

516. To the extent Da Rocha does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Da Rocha in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Da Rocha Anticipation Claims 1, 6; Da Rocha Combination Claim 1; Ex. B-6, claim 6.

517. Da Rocha discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Da Rocha Anticipation Claims 1, 2. It would have

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

518. To the extent this limitation is not expressly disclosed by Da Rocha, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Da Rocha discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

519. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Da Rocha's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

520. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Da Rocha to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Da Rocha's

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

521. Da Rocha discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* Da Rocha Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

522. Da Rocha discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Da Rocha Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

523. Da Rocha discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Da Rocha Anticipation Claim 7. To the extent Da Rocha does not by itself anticipate this claim element, Da Rocha in combination with Cantrill and/or Berry discloses this claim element. *See* Da Rocha Anticipation Claim 7; *see* Da Rocha Combination Claim 1; Ex. B-6, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

524. Da Rocha discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Da Rocha Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

525. Da Rocha discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Da Rocha Anticipation Claim 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

526. Da Rocha discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Da Rocha Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

527. Da Rocha discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Da Rocha Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

528. Da Rocha discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Da Rocha Anticipation Claim 8. To the extent Da Rocha does not by itself anticipate this claim element, Da Rocha in combination with Buchanan and/or Cantrill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses this claim element. *See* Da Rocha Anticipation Claim 8; *see* Da Rocha Combination Claim 2; Ex. B-6, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

529. Da Rocha discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Da Rocha Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

530. Da Rocha discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Da Rocha Anticipation Claim 13.

- b. *a processor;*

531. Da Rocha discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Da Rocha Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

532. Da Rocha discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Da Rocha Anticipation Claim 13.

- d. *running at least one thread in a first runtime environment;*

533. Da Rocha discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Da Rocha Anticipation Claim 13.

- e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

534. Da Rocha discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Da Rocha Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

535. Da Rocha discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Da Rocha Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

536. Da Rocha discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Da Rocha Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

537. Da Rocha discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Da Rocha Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

538. Da Rocha discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Da Rocha Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

539. Da Rocha discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

540. Da Rocha discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Da Rocha Anticipation Claim 14. To the extent Da Rocha does not by itself anticipate this claim element, Da Rocha in combination with Buchanan and/or Cantrill discloses this claim element. *See* Da Rocha Anticipation Claim 14; *see* Da Rocha Combination Claim 2; Ex. B-6, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

541. Da Rocha discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Da Rocha Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprises using Dijkstra's Self Stabilization Algorithm.*

542. Da Rocha discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Da Rocha does not by itself anticipate this claim element, Da Rocha in combination with RFC 2328 and/or Afek discloses this claim element. *See* Da Rocha Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-6, claim 18.

**L. Castelli in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek**

**1. *Motivation to Combine Castelli with Buchanan, Cantrill, and/or Berry***

543. Castelli in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Castelli discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Castelli and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill’s structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Castelli because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry’s systems enabled more operating parameters to



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Castelli are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Castelli to include these elements, rendering them obvious.

2. ***Motivation to Combine Castelli with RFC 2328 and/or Afek***

544. Castelli in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Castelli discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Castelli and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Castelli because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Castelli are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Castelli to include these elements, rendering them obvious.

3. ***Motivation to Combine Castelli with Turek***

545. Castelli in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Castelli and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Castelli and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Castelli because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Castelli in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Castelli and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Castelli and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

546. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

547. Castelli discloses this claim element. '659 Pat. at Cl. 1. *See* Castelli Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

548. Castelli discloses this claim element. '659 Pat. at Cl. 1. *See* Castelli Anticipation Claim 1..

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

549. Castelli discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Castelli Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

550. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

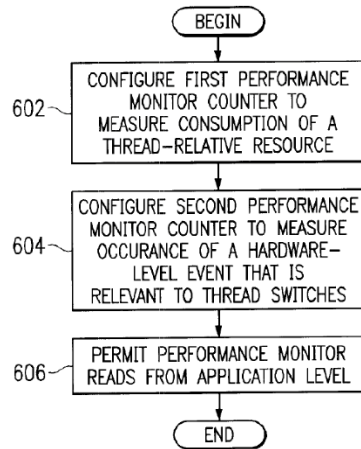
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Castelli, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

551. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

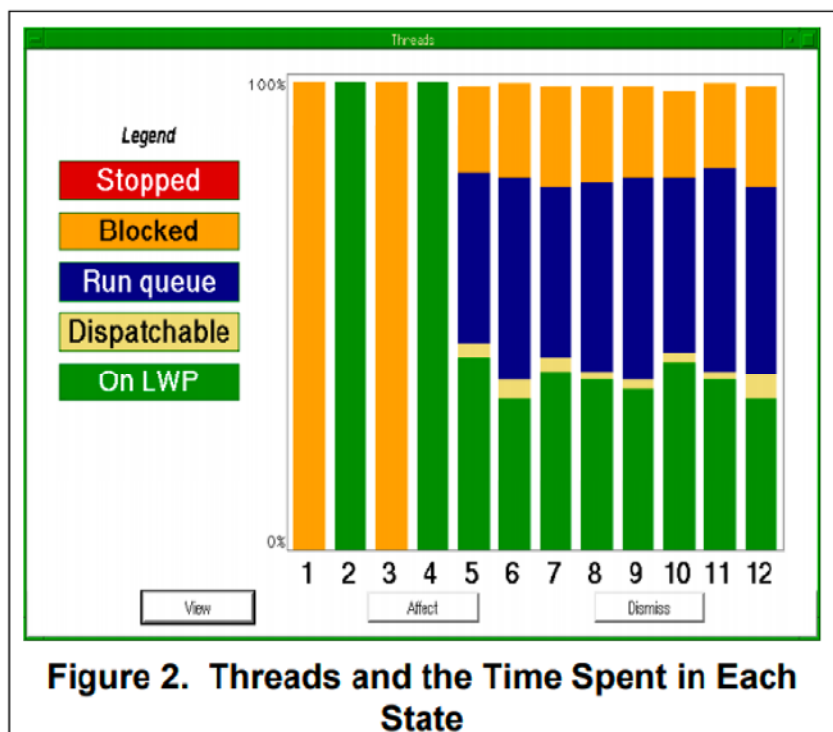
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors-any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult-it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Castelli to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

552. Castelli discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Castelli Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

553. Castelli discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Castelli Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

554. Castelli discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Castelli Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

555. Castelli discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Castelli Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

556. Castelli discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Castelli Anticipation Claim 1; Ex. B-12, claim 1; Turek Anticipation Claim 1.

**5. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

557. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-12, claim 2.

558. Castelli discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Castelli Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

559. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Castelli does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

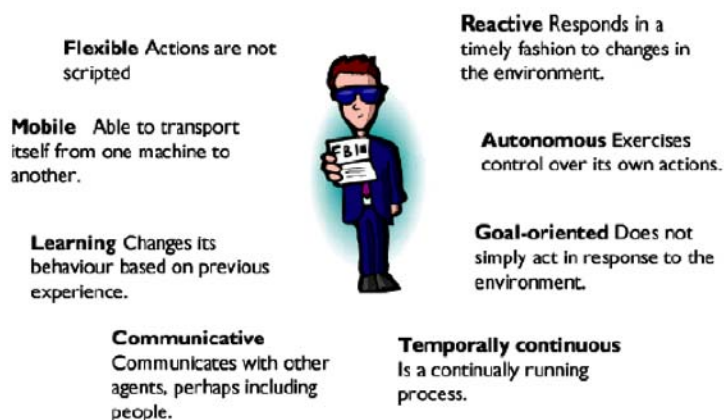
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

560. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*" Buchanan at sec. 4.

561. As described above, Cantrill also discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1; *see also* Castelli Combination Claim 1.

562. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Castelli to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

563. Castelli discloses claim 3, "[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds." '659 Pat. at Cl. 3. *See* Castelli Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

564. Castelli discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. *See* Castelli Anticipation Claim 6.

565. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

566. To the extent Castelli does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Castelli in

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

combination with RFC 2328 and/or Afek discloses this claim element. '659 Pat. at Cl. 1; *see* Castelli Anticipation Claims 1, 6; Castelli Combination Claim 1; Ex. B-12, claim 6.

567. Castelli discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Castelli Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

568. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Castelli discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

569. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Castelli's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

570. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Castelli to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Castelli's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

571. Castelli discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Castelli Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

572. Castelli discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Castelli Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

573. Castelli discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Castelli Anticipation Claim 7. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Cantrill and/or Berry discloses this claim element. *See* Castelli Anticipation Claim 7; *see* Castelli Combination Claim 1; Ex. B-12, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

574. Castelli discloses this element of claim 7, "detecting if an abnormality exists based on the monitored operational parameters," at least for all the reasons explained above regarding claim 1, element d. *See* Castelli Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

575. Castelli discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Castelli Anticipation Claim 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

576. Castelli discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Castelli Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

577. Castelli discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Castelli Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

578. Castelli discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Castelli Anticipation Claim 8. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Buchanan and/or Cantrill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses this claim element. *See* Castelli Anticipation Claim 8; *see* Castelli Combination Claim 2; Ex. B-12, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

579. Castelli discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Castelli Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

580. Castelli discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Castelli Anticipation Claim 13.

- b. *a processor;*

581. Castelli discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Castelli Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

582. Castelli discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Castelli Anticipation Claim 13.

- d. *running at least one thread in a first runtime environment;*

583. Castelli discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Castelli Anticipation Claim 13.

- e. *monitoring operational parameters relating to the or each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for each thread;*

584. Castelli discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Castelli Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

585. Castelli discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Castelli Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

586. Castelli discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Castelli Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

587. Castelli discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Castelli Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

588. Castelli discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Castelli Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

589. Castelli discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

590. Castelli discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Castelli Anticipation Claim 14. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Buchanan and/or Cantrill discloses this claim element. *See* Castelli Anticipation Claim 14; *see* Castelli Combination Claim 2; Ex. B-12, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

591. Castelli discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Castelli Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

592. Castelli discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

reasons explained above regarding claim 6. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with RFC 2328 and/or Afek discloses this claim element. *See* Castelli Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-12, claim 18.

**M. Conti in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek**

**1. *Motivation to Combine Conti with Buchanan, Cantrill, and/or Berry***

593. Conti in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Conti discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Conti and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Conti because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

extent that Buchanan, Cantrill, and/or Berry and Conti are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Conti to include these elements, rendering them obvious.

**2. *Motivation to Combine Conti with RFC 2328 and/or Afek***

594. Conti in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Conti discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Conti and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Conti because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Conti are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Conti to include these elements, rendering them obvious.

**3. *Motivation to Combine Conti with Turek***

595. Conti in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Conti and Turek disclose using mobile

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Conti and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Conti because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Conti in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Conti and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Conti and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

596. Conti discloses this claim element. *See* Conti Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

597. Conti discloses this claim element. '659 Pat. at Cl. 1. *See* Conti Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

598. Conti discloses this claim element. '659 Pat. at Cl. 1. *See* Conti Anticipation Claim 1..

599. Conti discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Conti Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

600. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

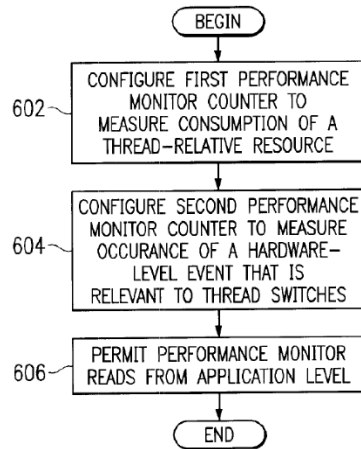
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Conti, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

601. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

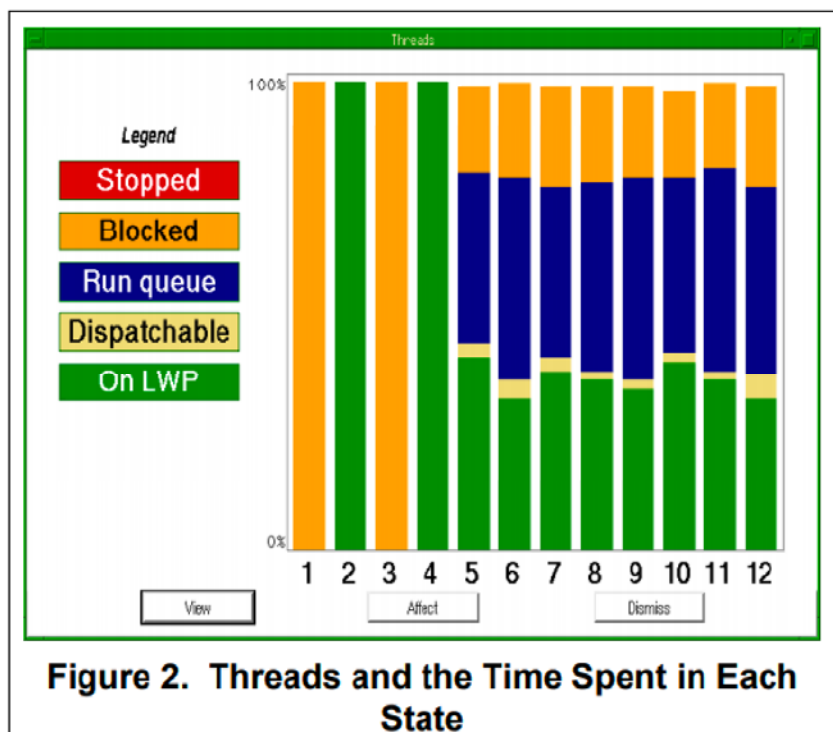
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Conti to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

602. Conti discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Conti Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

603. Conti discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Conti Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

604. Conti discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Conti Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

605. Conti discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Conti Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

606. Conti discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Conti Anticipation Claim 1; Ex. B-17, claim 1; Turek Anticipation Claim 1.

**5. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

607. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-17, claim 2.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

608. Conti discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Conti Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

609. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Conti does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. **It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.*

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-coordinating threads relatively simple.*** The main advantages of threads are:

o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

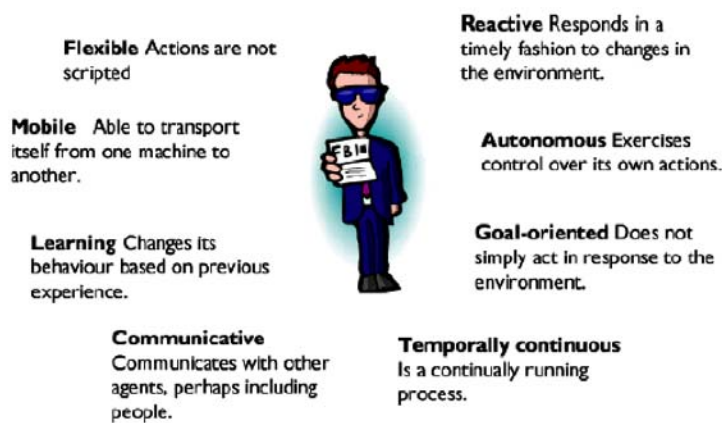
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

610. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

611. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Conti Combination Claim 1.

612. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Conti to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

613. Conti discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Conti Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

614. Conti discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Conti Anticipation Claim 6.

615. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

616. To the extent Conti does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Conti in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Conti Anticipation Claims 1, 6; Conti Combination Claim 1; Ex. B-17, claim 6.

617. Conti discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Conti Anticipation Claims 1, 2. It would have been

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

618. To the extent this limitation is not expressly disclosed by Conti, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. See NetFuel's Infringement Contentions for '659 Patent. Conti discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

619. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Conti's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

620. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Conti to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Conti's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. *Claim 7*

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

621. Conti discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* Conti Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

622. Conti discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Conti Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

623. Conti discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Conti Anticipation Claim 7. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Cantrill and/or Berry discloses this claim element. *See* Conti Anticipation Claim 7; *see* Conti Combination Claim 1; Ex. B-17, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

624. Conti discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Conti Anticipation Claim 7.

- e. *performing a corrective action to fix any detected abnormalities;*

625. Conti discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Conti Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

626. Conti discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Conti Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

627. Conti discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Conti Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

628. Conti discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Conti Anticipation Claim 8. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Buchanan and/or Cantrill discloses this claim element. *See* Conti Anticipation Claim 8; *see* Conti Combination Claim 2; Ex. B-17, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters to known thresholds.*

629. Conti discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Conti Anticipation Claim 9.

11. ***Claim 13***

a. *A system, comprising:*

630. Conti discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Conti Anticipation Claim 13.

b. *a processor;*

631. Conti discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Conti Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

632. Conti discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Conti Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

633. Conti discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Conti Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

634. Conti discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Conti Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

635. Conti discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Conti Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

636. Conti discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Conti Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

637. Conti discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Conti Anticipation Claim 13.

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

638. Conti discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Conti Anticipation Claim 13.

j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

639. Conti discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

640. Conti discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Conti Anticipation Claim 14. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Buchanan and/or Cantrill discloses this claim element. *See* Conti Anticipation Claim 14; *see* Conti Combination Claim 2; Ex. B-17, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

641. Conti discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Conti Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

642. Conti discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Conti does not by itself anticipate this claim element, Conti in combination with RFC 2328 and/or Afek discloses this claim element. *See* Conti Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-17, claim 18.

**N. NetRanger in combination with Kasteleijn/Turek/Goldman and/or RFC**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2328/Afek and/or Turek****1. *Motivation to Combine NetRanger with Buchanan, Cantrill, and/or Berry***

643. NetRanger in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. NetRanger discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of NetRanger and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in NetRanger because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and NetRanger are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or NetRanger to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. *Motivation to Combine NetRanger with RFC 2328 and/or Afek***

644. NetRanger in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. NetRanger discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of NetRanger and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in NetRanger because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and NetRanger are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or NetRanger to include these elements, rendering them obvious.

**3. *Motivation to Combine NetRanger with Turek***

645. NetRanger in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both NetRanger and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

in the art to combine the teachings of NetRanger and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in NetRanger because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in NetRanger in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that NetRanger and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify NetRanger and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

646. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

647. NetRanger discloses this claim element. '659 Pat. at Cl. 1. *See* NetRanger Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

648. NetRanger discloses this claim element. '659 Pat. at Cl. 1. *See* NetRanger Anticipation Claim 1..

649. NetRanger discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* NetRanger Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

650. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

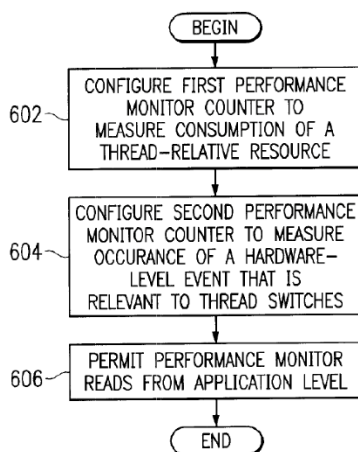


FIG. 6A

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with NetRanger, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

651. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

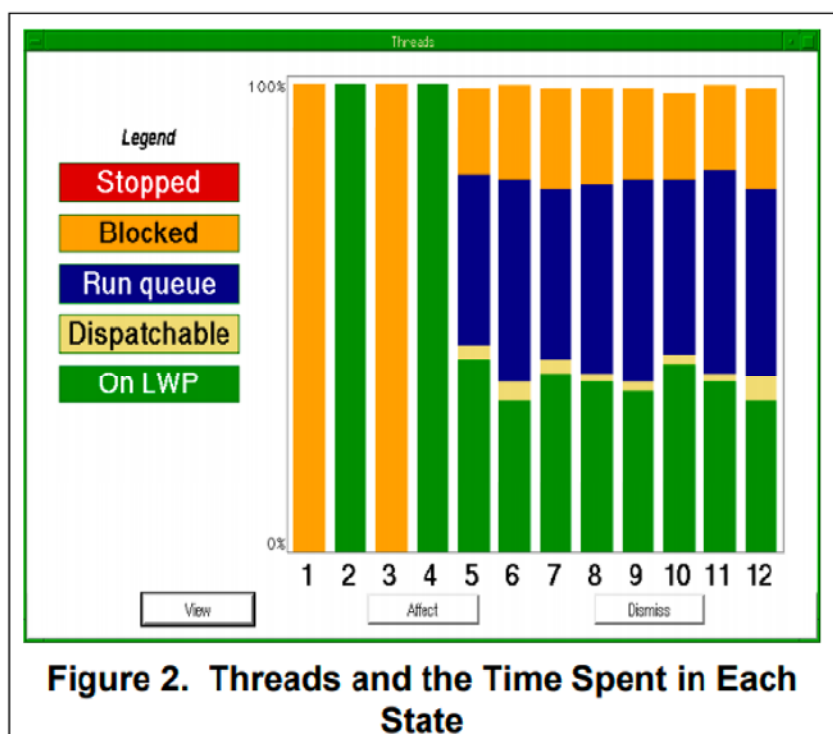
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler’s job is to ensure that enough computation tasks are performed that sufficient detail is

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors-any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult-it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify NetRanger to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *detecting if there is an abnormality in the monitored operational parameters;*

652. NetRanger discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* NetRanger Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

653. NetRanger discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* NetRanger Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

654. NetRanger discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* NetRanger Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

655. NetRanger discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* NetRanger Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

656. NetRanger discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* NetRanger Anticipation Claim 1; Ex. B-8, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

657. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-8, claim 2.

658. NetRanger discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* NetRanger Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

659. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as NetRanger does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.* To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

*Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).* A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). *These threads allow for smoother operation.* A server application that could only handle a request from one client would be of limited use. *Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

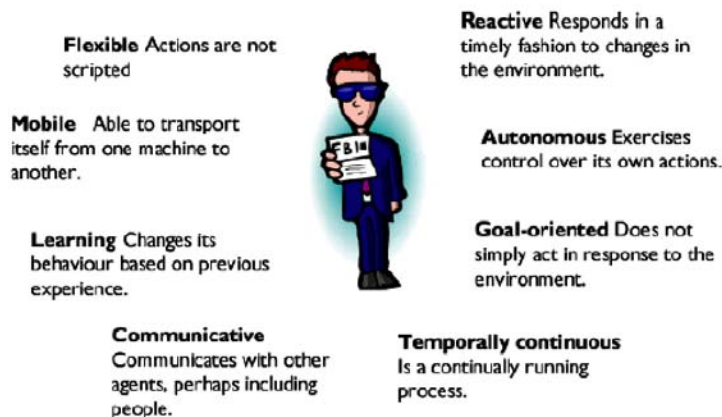
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1** Agent properties

Buchanan at Fig. 1.

660. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

661. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* NetRanger Combination Claim 1.

662. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify NetRanger to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. **Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters to known thresholds.*

663. NetRanger discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* NetRanger Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

664. NetRanger discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* NetRanger Anticipation Claim 6.

665. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

666. To the extent NetRanger does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, NetRanger in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* NetRanger Anticipation Claims 1, 6; NetRanger Combination Claim 1; Ex. B-8, claim 6.

667. NetRanger discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* NetRanger Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

668. To the extent this limitation is not expressly disclosed by NetRanger, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’659 Patent. NetRanger discloses using agents to detect and correct faults in a

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

669. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in NetRanger's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

670. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify NetRanger to create a corrective policy using Dijkstra's Self Stabilization Algorithm in NetRanger's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

671. NetRanger discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* NetRanger Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

672. NetRanger discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* NetRanger Anticipation Claim 7.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

673. NetRanger discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* NetRanger Anticipation Claim 7. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Cantrill and/or Berry discloses this claim element. *See* NetRanger Anticipation Claim 7; *see* NetRanger Combination Claim 1; Ex. B-8, claim 7.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

674. NetRanger discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* NetRanger Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

675. NetRanger discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* NetRanger Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

676. NetRanger discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* NetRanger Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

677. NetRanger discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* NetRanger Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

678. NetRanger discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* NetRanger Anticipation Claim 8. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Buchanan and/or Cantrill discloses this claim element. *See* NetRanger Anticipation Claim 8; *see* NetRanger Combination Claim 2; Ex. B-8, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

679. NetRanger discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* NetRanger Anticipation Claim 9.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. *Claim 13*

a. *A system, comprising:*

680. NetRanger discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* NetRanger Anticipation Claim 13.

b. *a processor;*

681. NetRanger discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* NetRanger Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

682. NetRanger discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* NetRanger Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

683. NetRanger discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* NetRanger Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

684. NetRanger discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* NetRanger Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

685. NetRanger discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* NetRanger Anticipation Claim 13.

g. *and performing a corrective action to fix any detected*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

686. NetRanger discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* NetRanger Anticipation Claim 13.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

687. NetRanger discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* NetRanger Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

688. NetRanger discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* NetRanger Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

689. NetRanger discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

690. NetRanger discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* NetRanger Anticipation Claim 14. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Buchanan and/or Cantrill discloses this claim element. *See* NetRanger Anticipation Claim 14; *see* NetRanger Combination Claim 2; Ex. B-8, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

691. NetRanger discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* NetRanger Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

692. NetRanger discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with RFC 2328 and/or Afek discloses this claim element. *See* NetRanger Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-8, claim 18.

**O. NSMIA in combination with Turek, Buchanan/Cantrill/Berry, and/or RFC**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2328/Afek and/or Turek****1. *Motivation to Combine NSMIA with Buchanan, Cantrill, and/or Berry***

693. NSMIA in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. NSMIA discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of NSMIA and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in NSMIA because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and NSMIA are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or NSMIA to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. *Motivation to Combine NSMIA with RFC 2328 and/or Afek***

694. NSMIA in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. NSMIA discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of NSMIA and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in NSMIA because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and NSMIA are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or NSMIA to include these elements, rendering them obvious.

**3. *Motivation to Combine NSMIA with Turek***

695. NSMIA in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both NSMIA and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of NSMIA and Turek. For example, a person of ordinary skill

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

in the art would have been motivated to implement Turek's multithreading in NSMIA because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in NSMIA in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that NSMIA and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify NSMIA and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

696. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

697. NSMIA discloses this claim element. '659 Pat. at Cl. 1. *See* NSMIA Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

698. NSMIA discloses this claim element. '659 Pat. at Cl. 1. *See* NSMIA Anticipation Claim 1.

699. NSMIA discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* NSMIA Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

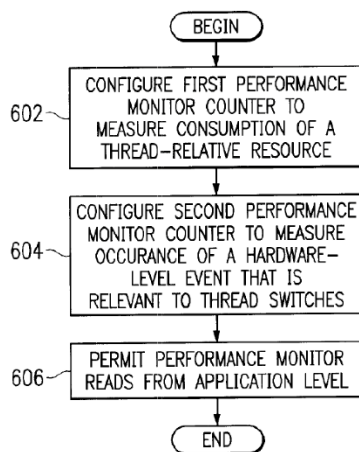
700. Berry discloses "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread." '659 Pat., Cl. 1. For example, Berry

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



*FIG. 6A*

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

provides further explicit motivation to combine with NSMIA , explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

701. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing*

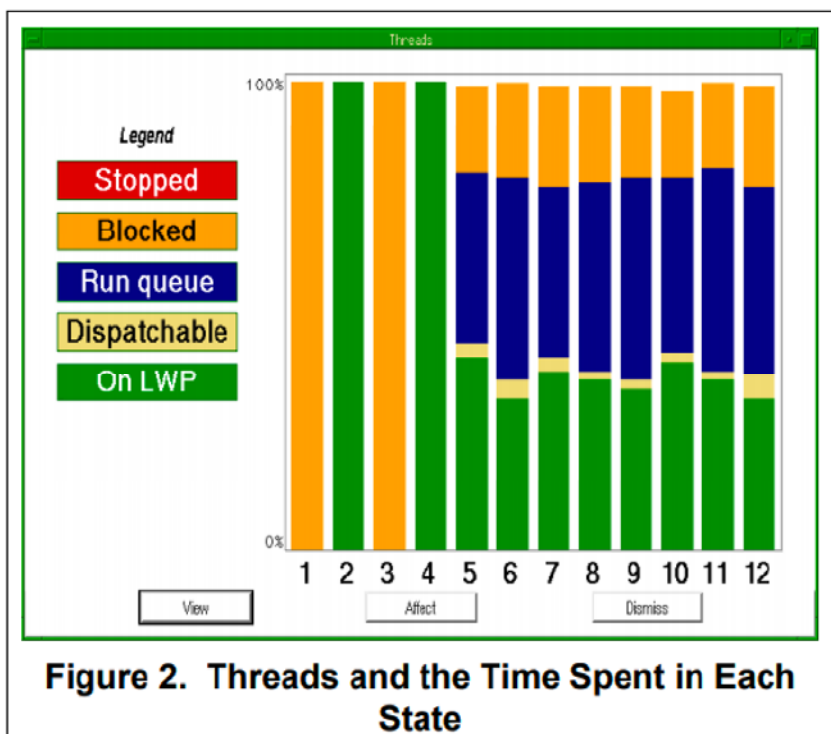


**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult-it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify NSMIA to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*parameters;*

702. NSMIA discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* NSMIA Anticipation Claim 1.

e. *and performing a corrective action to fix any detected abnormalities,*

703. NSMIA discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* NSMIA Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

704. NSMIA discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* NSMIA Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

705. NSMIA discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* NSMIA Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

706. NSMIA discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent NSMIA does not by itself anticipate this claim element, NSMIA in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* NSMIA Anticipation Claim 1; Ex. B-19, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

707. To the extent NSMIA does not by itself anticipate this claim element, NSMIA in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-19, claim 2.

708. NSMIA discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* NSMIA Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

709. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as NSMIA does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.* To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

*Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).* A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). *These threads allow for smoother operation.* A server application that could only handle a request from one client would be of limited use. *Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

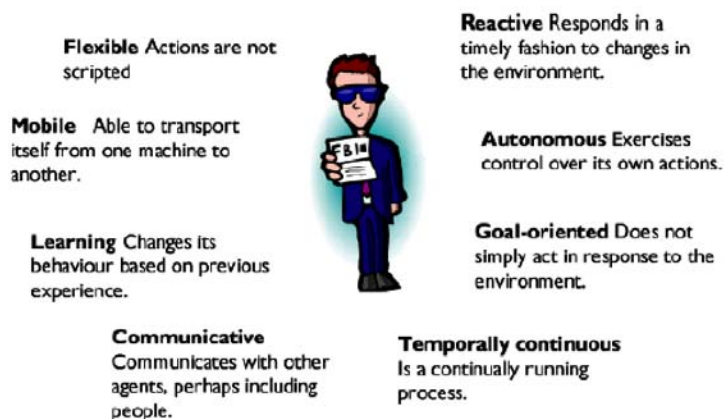
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1** Agent properties

Buchanan at Fig. 1.

710. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

711. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* NSMIA Combination Claim 1.

712. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify NSMIA to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. **Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

713. NSMIA discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* NSMIA Anticipation Claim 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

714. NSMIA discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* NSMIA Anticipation Claim 6.

715. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

716. To the extent NSMIA does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, NSMIA in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* NSMIA Anticipation Claims 1, 6; NSMIA Combination Claim 1; Ex. B-19, claim 6.

717. NSMIA discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* NSMIA Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

718. To the extent this limitation is not expressly disclosed by NSMIA , it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for ’659 Patent. NSMIA discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

719. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in NSMIA 's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

720. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify NSMIA to create a corrective policy using Dijkstra's Self Stabilization Algorithm in NSMIA 's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. **Claim 7**

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

721. NSMIA discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. See NSMIA Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *running at least one thread in a first runtime environment;*

722. NSMIA discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* NSMIA Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

723. NSMIA discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* NSMIA Anticipation Claim 7. To the extent NSMIA does not by itself anticipate this claim element, NSMIA in combination with Cantrill and/or Berry discloses this claim element. *See* NSMIA Anticipation Claim 7; *see* NSMIA Combination Claim 1; Ex. B-19, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

724. NSMIA discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* NSMIA Anticipation Claim 7.

- e. *performing a corrective action to fix any detected abnormalities;*

725. NSMIA discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* NSMIA Anticipation Claim 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

726. NSMIA discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* NSMIA Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

727. NSMIA discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* NSMIA Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

728. NSMIA discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* NSMIA Anticipation Claim 8. To the extent NSMIA does not by itself anticipate this claim element, NSMIA in combination with Buchanan and/or Cantrill discloses this claim element. *See* NSMIA Anticipation Claim 8; *see* NSMIA Combination Claim 2; Ex. B-19, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

729. NSMIA discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* NSMIA Anticipation Claim 9.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 13***

a. *A system, comprising:*

730. NSMIA discloses the preamble of claim 13, “[a] system, comprising . . .” ’659 Pat. at Cl. 13. *See* NSMIA Anticipation Claim 13.

b. *a processor;*

731. NSMIA discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* NSMIA Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

732. NSMIA discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* NSMIA Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

733. NSMIA discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* NSMIA Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

734. NSMIA discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* NSMIA Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

735. NSMIA discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* NSMIA Anticipation Claim 13.

g. *and performing a corrective action to fix any detected*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*abnormalities;*

736. NSMIA discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* NSMIA Anticipation Claim 13.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

737. NSMIA discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* NSMIA Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

738. NSMIA discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* NSMIA Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

739. NSMIA discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

740. NSMIA discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* NSMIA Anticipation Claim 14. To the extent NSMIA does not by itself anticipate this claim element, NSMIA in combination with Buchanan and/or Cantrill discloses this claim element. *See* NSMIA Anticipation Claim 14; *see* NSMIA Combination Claim 2; Ex. B-19, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

741. NSMIA discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* NSMIA Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

742. NSMIA discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent NSMIA does not by itself anticipate this claim element, NSMIA in combination with RFC 2328 and/or Afek discloses this claim element. *See* S Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-19, claim 18.

**P. [REDACTED] in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine [REDACTED] with***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY*****Buchanan, Cantrill, and/or Berry***

743. [REDACTED] in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. [REDACTED] discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of [REDACTED] and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in [REDACTED] because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and [REDACTED] are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or [REDACTED] to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**2. ***Motivation to Combine [REDACTED] with RFC 2328 and/or Afek***

744. [REDACTED] in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. [REDACTED] discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of [REDACTED] and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in [REDACTED] because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and [REDACTED] are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or [REDACTED] to include these elements, rendering them obvious.

3. ***Motivation to Combine [REDACTED] with Turek***

745. [REDACTED] in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both [REDACTED] and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of [REDACTED] and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in [REDACTED] because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in [REDACTED] in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that [REDACTED] and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify [REDACTED] and/or Turek to include these elements, rendering them obvious.

4. *Claim 1*

a. *A computer-implemented method, comprising:*

746. [REDACTED] discloses this claim element. See [REDACTED]  
[REDACTED] Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

747. [REDACTED] discloses this claim element. '659 Pat. at Cl.  
1. See [REDACTED] Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

748. [REDACTED] discloses this claim element. '659 Pat. at Cl.  
1. See [REDACTED] Claim 1..

749. [REDACTED] discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. See

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

750. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

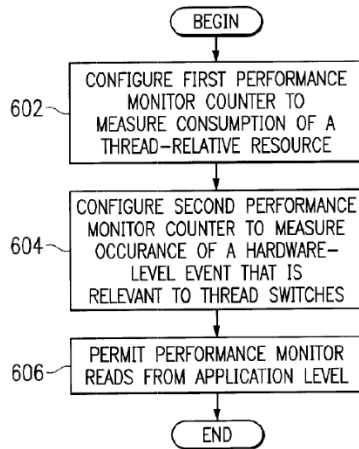
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Cisco Receive Adjacency Protection, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

751. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

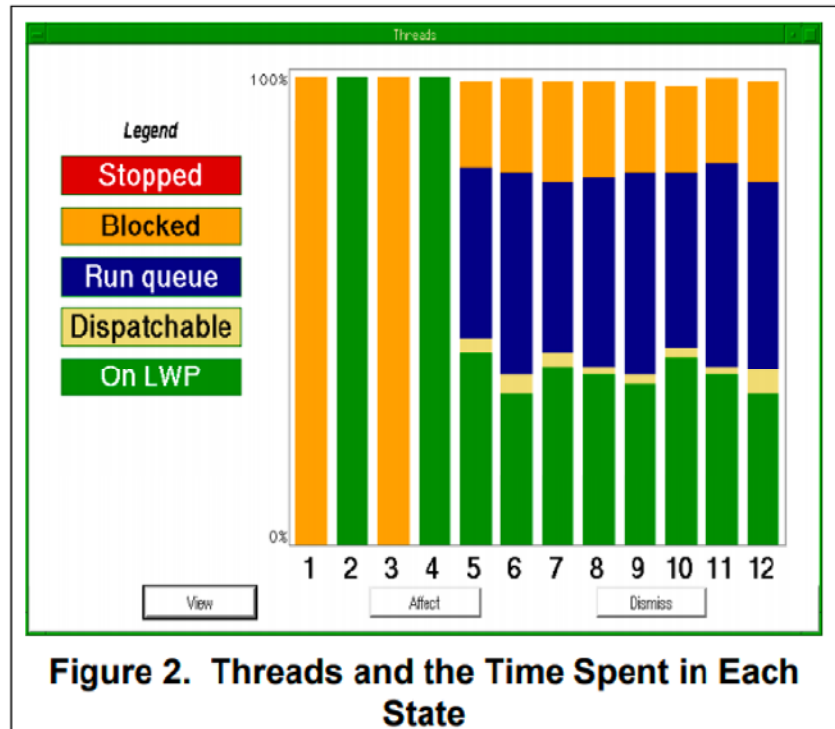
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify [REDACTED] to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

752. [REDACTED] discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. See [REDACTED] Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

753. [REDACTED] discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. See [REDACTED] Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. See Ex. B-1 at limitation 1.5. A person of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

754. [REDACTED] discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl.

1. *See* [REDACTED] Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

755. [REDACTED] discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* [REDACTED]

Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

756. [REDACTED] discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Turek discloses this claim element. ’659

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Pat. at Cl. 1; *see* [REDACTED] Claim 1; Ex. B-10, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

757. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-1, claim 2.

758. [REDACTED] discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* [REDACTED] Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

759. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as [REDACTED] does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

*Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).* A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). *These threads allow for smoother operation.* A server application that could only handle a request from one client would be of limited use. *Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

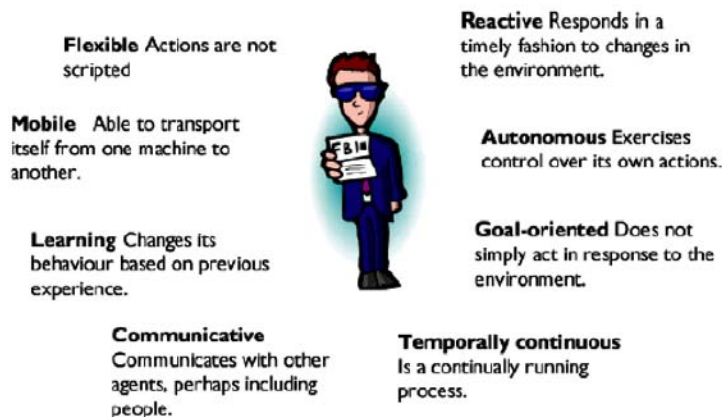
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1** Agent properties

Buchanan at Fig. 1.

760. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

761. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* [REDACTED] Combination Claim 1.

762. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify [REDACTED] to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. **Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters to known thresholds.*

763. [REDACTED] discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* [REDACTED] Anticipation Claim 3.

7. **Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

764. [REDACTED] on discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* [REDACTED] Anticipation Claim 6.

765. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

766. To the extent [REDACTED] does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, [REDACTED] in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* [REDACTED] Anticipation Claims 1, 6; [REDACTED] Combination Claim 1; Ex. B-10, claim 6.

767. [REDACTED] tection discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* [REDACTED] Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

768. To the extent this limitation is not expressly disclosed by [REDACTED] [REDACTED] it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. [REDACTED] discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

769. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in [REDACTED]'s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

770. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify [REDACTED] [REDACTED] to create a corrective policy using Dijkstra's Self Stabilization Algorithm in [REDACTED] multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. *Claim 7*

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

771. [REDACTED] discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* [REDACTED] Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

772. [REDACTED] discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* [REDACTED] Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

773. [REDACTED] discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* [REDACTED] Anticipation Claim 7. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Cantrill and/or Berry discloses this claim element. *See* [REDACTED] Claim 7; *see* [REDACTED] Combination Claim 1; Ex. B-10, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

774. [REDACTED] discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* [REDACTED] Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

775. [REDACTED] discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* [REDACTED] Anticipation Claim 7.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

776. [REDACTED] discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* [REDACTED] Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

777. [REDACTED] discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* [REDACTED] Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

778. [REDACTED] discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* [REDACTED]

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Anticipation Claim 8. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Buchanan and/or Cantrill discloses this claim element. See [REDACTED] Anticipation Claim 8; see [REDACTED] Combination Claim 2; Ex. B-10, claim 8.

10. **Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

779. [REDACTED] discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. See [REDACTED] Claim 9.

11. **Claim 13**

- a. *A system, comprising:*

780. [REDACTED] discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. See [REDACTED] Anticipation Claim 13.

- b. *a processor;*

781. [REDACTED] discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. See [REDACTED] Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

782. [REDACTED] discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. See [REDACTED] Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *running at least one thread in a first runtime environment;*

783. [REDACTED] discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. See [REDACTED] Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

784. [REDACTED] discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. See [REDACTED] Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

785. [REDACTED] discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. See [REDACTED] Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

786. [REDACTED] discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. See [REDACTED] Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

787. [REDACTED] discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the reasons explained above regarding claim 1, element f. *See* [REDACTED]

Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

788. [REDACTED] discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* [REDACTED] Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

789. [REDACTED] discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

790. [REDACTED] discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* [REDACTED] Anticipation Claim 14. To the extent Cisco Receive Adjacency Protection does not by itself anticipate this claim element, [REDACTED] in combination with Buchanan and/or Cantrill discloses this claim element. *See* [REDACTED] Anticipation Claim 14; *see* [REDACTED] Combination Claim 2; Ex. B-10, claim 14.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

791. [REDACTED] discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. See [REDACTED] Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

792. [REDACTED] discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] combination with RFC 2328 and/or Afek discloses this claim element. See [REDACTED] Claim 18; see Da Rocha Combination Claim 6; Ex. B-10, claim 18.

**Q. Cisco Catalyst 5000 in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine Cisco Catalyst 5000R with Buchanan, Cantrill, and/or Berry***

793. Cisco Catalyst 5000 in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco Catalyst 5000 discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco Catalyst 5000 and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Cisco Catalyst 5000 because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Cisco Catalyst 5000 are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Cisco Catalyst 5000 to include these elements, rendering them obvious.

2. ***Motivation to Combine Cisco Catalyst 5000 with RFC 2328 and/or Afek***

794. Cisco Catalyst 5000 in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco Catalyst 5000 discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management.. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

teachings of Cisco Catalyst 5000 and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Cisco Catalyst 5000 because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Cisco Catalyst 5000 are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Cisco Catalyst 5000 to include these elements, rendering them obvious.

3. ***Motivation to Combine Cisco Catalyst 5000 with Turek***

795. Cisco Catalyst 5000 in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Cisco Catalyst 5000 and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco Catalyst 5000 and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Cisco Catalyst 5000 because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Cisco Catalyst 5000 in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Cisco Catalyst 5000 and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

ordinary skill in the art, and it would have been obvious to modify Cisco Catalyst 5000 and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

796. Cisco Catalyst 5000 discloses this claim element. *See* Cisco Catalyst 5000 Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

797. Cisco Catalyst 5000 discloses this claim element. '659 Pat. at Cl. 1. *See* Cisco Catalyst 5000 Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

798. Cisco Catalyst 5000 discloses this claim element. '659 Pat. at Cl. 1. *See* Cisco Catalyst 5000 Anticipation Claim 1..

799. Cisco Catalyst 5000 discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Cisco Catalyst 5000 Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

800. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of

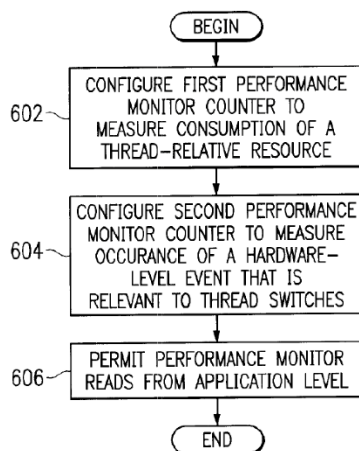
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64.

Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



*FIG. 6A*

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Cisco Catalyst 5000, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

801. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

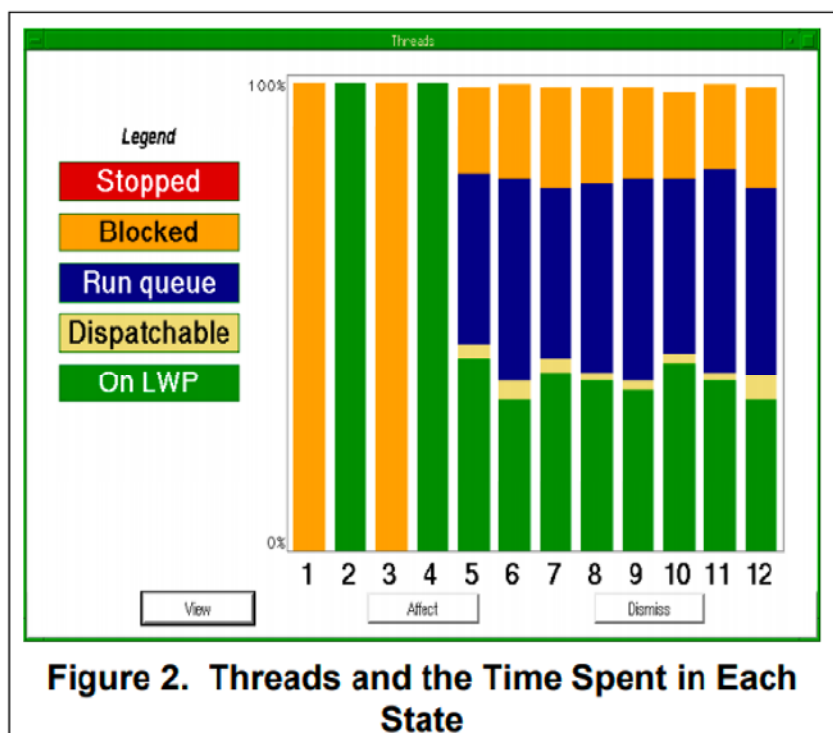
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Cisco Catalyst 5000 to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

802. Cisco Catalyst 5000 discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Cisco Catalyst 5000 Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

803. Cisco Catalyst 5000 discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Cisco Catalyst 5000 Anticipation

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek '070. See Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

804. Cisco Catalyst 5000 discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Cisco Catalyst 5000 Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

805. Cisco Catalyst 5000 discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Cisco Catalyst 5000 Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

806. Cisco Catalyst 5000 discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Cisco Catalyst 5000 does not by itself anticipate this claim element, Cisco Catalyst 5000 in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Cisco Catalyst 5000 Anticipation Claim 1; Ex. B-9, claim 1; Turek Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**5. *Claim 2*

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

807. To the extent Cisco Catalyst 5000 does not by itself anticipate this claim element, Cisco Catalyst 5000 in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-9, claim 2.

808. Cisco Catalyst 5000 discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Cisco Catalyst 5000 Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

809. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Cisco Catalyst 5000 does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.* There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. *It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.* To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

*Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).* A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). *These threads allow for smoother operation.* A server

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

application that could only handle a request from one client would be of limited use. *Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

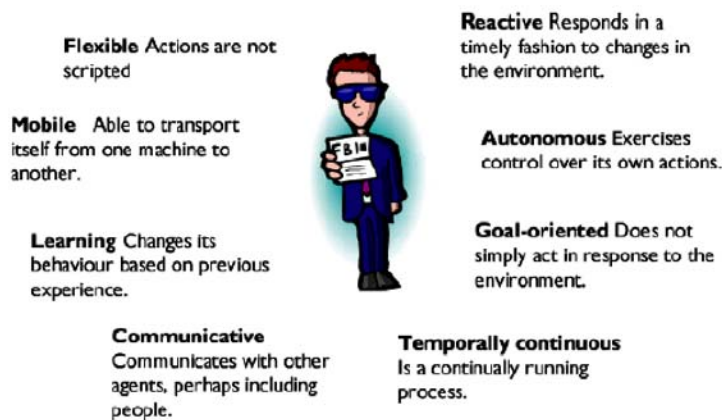
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

810. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*" Buchanan at sec. 4.

811. As described above, Cantrill also discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1; *see also* Cisco Catalyst 5000 Combination Claim 1.

812. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Cisco Catalyst 5000 to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

813. Cisco Catalyst 5000 discloses claim 3, "[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds." '659 Pat. at Cl. 3. *See* Cisco Catalyst 5000 Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

814. Cisco Catalyst 5000 discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. *See* Cisco Catalyst 5000 Anticipation Claim 6.

815. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

816. To the extent Cisco Catalyst 5000 does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Cisco Catalyst 5000 in combination with RFC 2328 and/or Afek discloses this claim element. '659 Pat. at Cl. 1; *see* Cisco Catalyst 5000 Anticipation Claims 1, 6; Cisco Catalyst 5000 Combination Claim 1; Ex. B-9, claim 6.

817. Cisco Catalyst 5000 discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Cisco Catalyst 5000 Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

818. To the extent this limitation is not expressly disclosed by Cisco Catalyst 5000, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Cisco Catalyst 5000 discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

819. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Cisco Catalyst 5000's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

820. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Cisco Catalyst 5000 to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Cisco Catalyst 5000's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

821. Cisco Catalyst 5000 discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Cisco Catalyst 5000 Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

822. Cisco Catalyst 5000 discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Cisco Catalyst 5000 Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

823. Cisco Catalyst 5000 discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Cisco Catalyst 5000 Anticipation Claim 7. To the extent Cisco Catalyst 5000 does not by itself anticipate this claim element, Cisco Catalyst 5000 in combination with Cantrill and/or Berry discloses this claim element. *See* Cisco Catalyst 5000 Anticipation Claim 7; *see* Cisco Catalyst 5000 Combination Claim 1; Ex. B-9, claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

824. Cisco Catalyst 5000 discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See Cisco Catalyst 5000 Anticipation Claim 7.*

- e. *performing a corrective action to fix any detected abnormalities;*

825. Cisco Catalyst 5000 discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Cisco Catalyst 5000 Anticipation Claim 7.*

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

826. Cisco Catalyst 5000 discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See Cisco Catalyst 5000 Anticipation Claim 7.*

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

827. Cisco Catalyst 5000 discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See Cisco Catalyst 5000 Anticipation Claim 7.*

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

828. Cisco Catalyst 5000 discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Cisco Catalyst 5000 Anticipation Claim 8. To the extent Cisco Catalyst 5000 does not by itself anticipate this claim element, Cisco Catalyst 5000 in combination with Buchanan and/or Cantrill discloses this claim element. *See* Cisco Catalyst 5000 Anticipation Claim 8; *see* Cisco Catalyst 5000 Combination Claim 2; Ex. B-9, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

829. Cisco Catalyst 5000 discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Cisco Catalyst 5000 Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

830. Cisco Catalyst 5000 discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Cisco Catalyst 5000 Anticipation Claim 13.

- b. *a processor;*

831. Cisco Catalyst 5000 discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Cisco Catalyst 5000 Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

832. Cisco Catalyst 5000 discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Cisco Catalyst 5000 Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *running at least one thread in a first runtime environment;*

833. Cisco Catalyst 5000 discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Cisco Catalyst 5000 Anticipation Claim 13.*

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

834. Cisco Catalyst 5000 discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See Cisco Catalyst 5000 Anticipation Claim 13.*

f. *detecting there is an abnormality in the monitored operational parameters;*

835. Cisco Catalyst 5000 discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See Cisco Catalyst 5000 Anticipation Claim 13.*

g. *and performing a corrective action to fix any detected abnormalities;*

836. Cisco Catalyst 5000 discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Cisco Catalyst 5000 Anticipation Claim 13.*

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

837. Cisco Catalyst 5000 discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See Cisco Catalyst 5000 Anticipation Claim 13.*

i. *wherein performing the corrective action is based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective policy applied by the agent running within the first runtime environment,*

838. Cisco Catalyst 5000 discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See Cisco Catalyst 5000 Anticipation Claim 13.*

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

839. Cisco Catalyst 5000 discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

840. Cisco Catalyst 5000 discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See Cisco Catalyst 5000 Anticipation Claim 14.* To the extent Cisco Catalyst 5000 does not by itself anticipate this claim element, Cisco Catalyst 5000 in combination with Buchanan and/or Cantrill discloses this claim element. *See Cisco Catalyst 5000 Anticipation Claim 14; see Cisco Catalyst 5000 Combination Claim 2; Ex. B-9, claim 14.*

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

841. Cisco Catalyst 5000 discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Cisco Catalyst 5000 Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

842. Cisco Catalyst 5000 discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Cisco Catalyst 5000 does not by itself anticipate this claim element, Cisco Catalyst 5000 in combination with RFC 2328 and/or Afek discloses this claim element. *See* Cisco Catalyst 5000 Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-9, claim 18.

**R. Cisco rACL in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine Cisco rACL with Buchanan, Cantrill, and/or Berry***

843. Cisco rACL in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco rACL discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco rACL and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill’s structure for monitoring numerous



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

operating parameters of a single core in the multicore system disclosed in Cisco rACL because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Cisco rACL are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Cisco rACL to include these elements, rendering them obvious.

**2. *Motivation to Combine Cisco rACL with RFC 2328 and/or Afek***

844. Cisco rACL in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco rACL discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco rACL and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Cisco rACL because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Cisco rACL are found to not include each element of the asserted claims of the '659 Patent,



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Cisco rACL to include these elements, rendering them obvious.

3. ***Motivation to Combine Cisco rACL with Turek***

845. Cisco rACL in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Cisco rACL and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco rACL and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Cisco rACL because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Cisco rACL in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Cisco rACL and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Cisco rACL and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

846. Cisco rACL discloses this claim element. *See* Cisco rACL Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *running at least one thread in a first runtime environment;*

847. Cisco rACL discloses this claim element. '659 Pat. at Cl. 1. *See* Cisco rACL Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

848. Cisco rACL discloses this claim element. '659 Pat. at Cl. 1. *See* Cisco rACL Anticipation Claim 1..

849. Cisco rACL discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Cisco rACL Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

850. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry's network performance monitoring system:

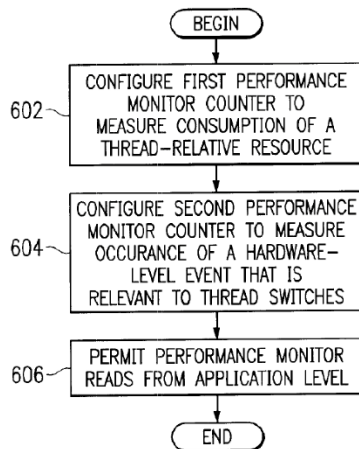


FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Cisco rACL, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

***Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.***

In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. ***Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.*** Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

851. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

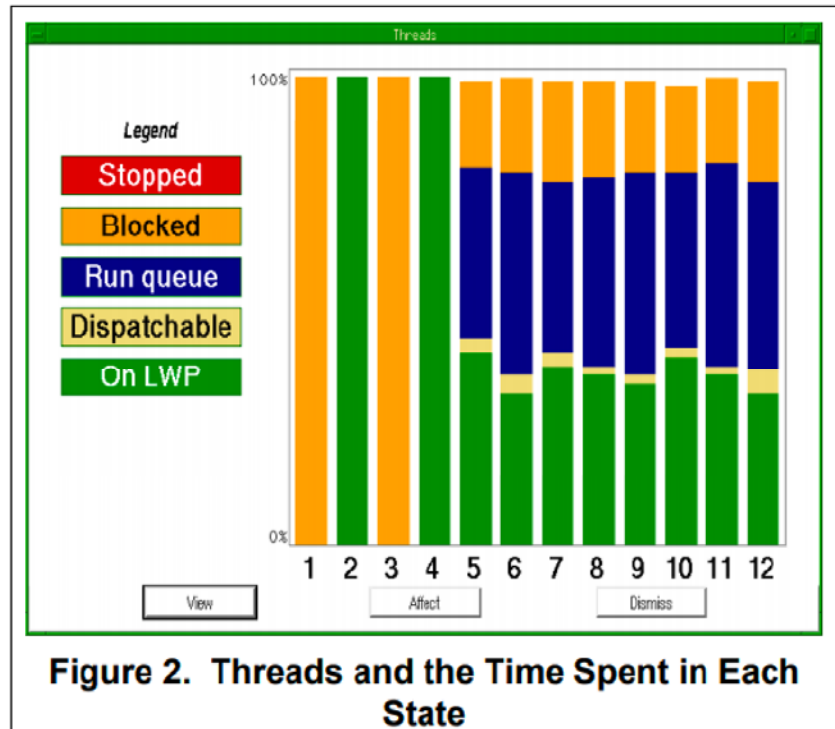
1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler’s job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2 (emphases added).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Cisco rACL to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

852. Cisco rACL discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Cisco rACL Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

853. Cisco rACL discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Cisco rACL Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

854. Cisco rACL discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Cisco rACL Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

855. Cisco rACL discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Cisco rACL Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

856. Cisco rACL discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Cisco rACL does not by itself anticipate this claim element, Cisco rACL in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Cisco rACL Anticipation Claim 1; Ex. B-11, claim 1; Turek Anticipation Claim 1.

5. **Claim 2**

- a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

857. To the extent Cisco rACL does not by itself anticipate this claim element, Cisco rACL in combination with Cantrill and/or Buchanan discloses this claim element. '659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-11, claim 2.

858. Cisco rACL discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Cisco rACL Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

859. Buchanan discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Cisco rACL does:

***Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.*** There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

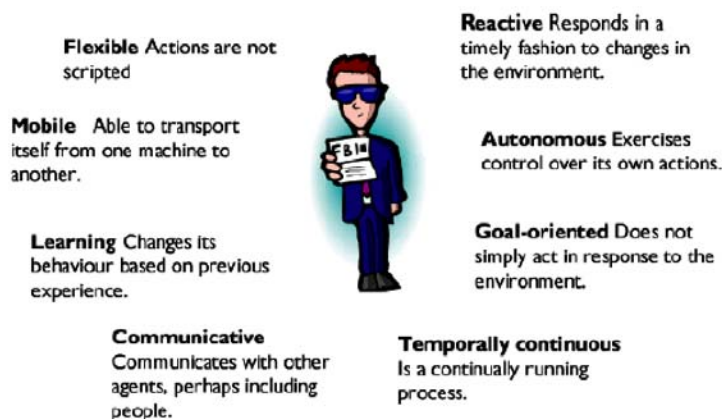
- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

860. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*" Buchanan at sec. 4.

861. As described above, Cantrill also discloses "wherein the corrective action comprises a load balancing operation." '659 Pat., Cl. 1; *see also* Cisco rACL Combination Claim 1.

862. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Cisco rACL to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. ***Claim 3***

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

863. Cisco rACL discloses claim 3, "[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds." '659 Pat. at Cl. 3. *See* Cisco rACL Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm.*

864. Cisco rACL discloses claim 6, "[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithm." '659 Pat. at Cl. 6. *See* Cisco rACL Anticipation Claim 6.

865. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

866. To the extent Cisco rACL does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Cisco rACL in

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

combination with RFC 2328 and/or Afek discloses this claim element. '659 Pat. at Cl. 1; *see* Cisco rACL Anticipation Claims 1, 6; Cisco rACL Combination Claim 1; Ex. B-11, claim 6.

867. Cisco rACL discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Cisco rACL Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

868. To the extent this limitation is not expressly disclosed by Cisco rACL, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. Cisco rACL discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

869. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Cisco rACL's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

870. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Cisco rACL to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Cisco rACL's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

871. Cisco rACL discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Cisco rACL Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

872. Cisco rACL discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Cisco rACL Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

873. Cisco rACL discloses this element of claim 7, "monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread," at least for all the reasons explained above regarding claim 1, element c. *See* Cisco rACL Anticipation Claim 7. To the extent Cisco rACL does not by itself anticipate this claim element, Cisco rACL in combination with Cantrill and/or Berry discloses this claim element. *See* Cisco rACL Anticipation Claim 7; *see* Cisco rACL Combination Claim 1; Ex. B-11, claim 7.

- d. *detecting if an abnormality exists based on the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters;*

874. Cisco rACL discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Cisco rACL Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

875. Cisco rACL discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Cisco rACL Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

876. Cisco rACL discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Cisco rACL Anticipation Claim 7.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

877. Cisco rACL discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Cisco rACL Anticipation Claim 7.

9. ***Claim 8***

a. *The computer-readable medium of claim 7, wherein the corrective*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*action comprises a load balancing operation.*

878. Cisco rACL discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Cisco rACL Anticipation Claim 8. To the extent Cisco rACL does not by itself anticipate this claim element, Cisco rACL in combination with Buchanan and/or Cantrill discloses this claim element. *See* Cisco rACL Anticipation Claim 8; *see* Cisco rACL Combination Claim 2; Ex. B-11, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

879. Cisco rACL discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Cisco rACL Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

880. Cisco rACL discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Cisco rACL Anticipation Claim 13.

- b. *a processor;*

881. Cisco rACL discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Cisco rACL Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

882. Cisco rACL discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Cisco rACL Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *running at least one thread in a first runtime environment;*

883. Cisco rACL discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Cisco rACL Anticipation Claim 13.*

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

884. Cisco rACL discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See Cisco rACL Anticipation Claim 13.*

f. *detecting there is an abnormality in the monitored operational parameters;*

885. Cisco rACL discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See Cisco rACL Anticipation Claim 13.*

g. *and performing a corrective action to fix any detected abnormalities;*

886. Cisco rACL discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Cisco rACL Anticipation Claim 13.*

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

887. Cisco rACL discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See Cisco rACL Anticipation Claim 13.*

i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*runtime environment,*

888. Cisco rACL discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Cisco rACL Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

889. Cisco rACL discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

890. Cisco rACL discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Cisco rACL Anticipation Claim 14. To the extent Cisco rACL does not by itself anticipate this claim element, Cisco rACL in combination with Buchanan and/or Cantrill discloses this claim element. *See* Cisco rACL Anticipation Claim 14; *see* Cisco rACL Combination Claim 2; Ex. B-11, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

891. Cisco rACL discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

thresholds,” at least for all the reasons explained above regarding claim 3. *See* Cisco rACL Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

892. Cisco rACL discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Cisco rACL does not by itself anticipate this claim element, Cisco rACL in combination with RFC 2328 and/or Afek discloses this claim element. *See* rACL Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-11, claim 18.

**S. IBM xSeries Server Software Rejuvenation Agent in combination with Kasteleijn/Turek/Goldman and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine IBM xSeries Server Software Rejuvenation AgentR with Buchanan, Cantrill, and/or Berry***

893. IBM xSeries Server Software Rejuvenation Agent in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. IBM xSeries Server Software Rejuvenation Agent discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of IBM xSeries Server Software Rejuvenation Agent and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in IBM xSeries Server Software Rejuvenation Agent because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and IBM xSeries Server Software Rejuvenation Agent are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or IBM xSeries Server Software Rejuvenation Agent to include these elements, rendering them obvious.

2. ***Motivation to Combine IBM xSeries Server Software Rejuvenation Agent with RFC 2328 and/or Afek***

894. IBM xSeries Server Software Rejuvenation Agent in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. IBM xSeries Server Software Rejuvenation Agent discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of IBM xSeries Server Software Rejuvenation Agent and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in IBM xSeries Server Software Rejuvenation Agent because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and IBM xSeries Server Software Rejuvenation Agent are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or IBM xSeries Server Software Rejuvenation Agent to include these elements, rendering them obvious.

3. ***Motivation to Combine IBM xSeries Server Software Rejuvenation Agent with Turek***

895. IBM xSeries Server Software Rejuvenation Agent in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both IBM xSeries Server Software Rejuvenation Agent and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of IBM xSeries Server Software Rejuvenation Agent and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in IBM xSeries Server Software Rejuvenation Agent because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in IBM xSeries Server Software Rejuvenation Agent in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

environments/across multiple agents. Additionally, to the extent that IBM xSeries Server Software Rejuvenation Agent and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify IBM xSeries Server Software Rejuvenation Agent and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

896. IBM xSeries Server Software Rejuvenation Agent discloses this claim element. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

897. IBM xSeries Server Software Rejuvenation Agent discloses this claim element. '659 Pat. at Cl. 1. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

898. IBM xSeries Server Software Rejuvenation Agent discloses this claim element. '659 Pat. at Cl. 1. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1..

899. IBM xSeries Server Software Rejuvenation Agent discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

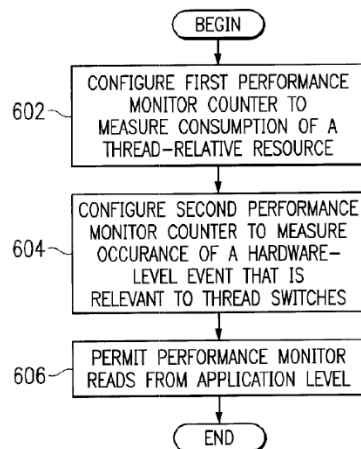
900. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



*FIG. 6A*

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with IBM xSeries Server Software Rejuvenation Agent, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

901. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

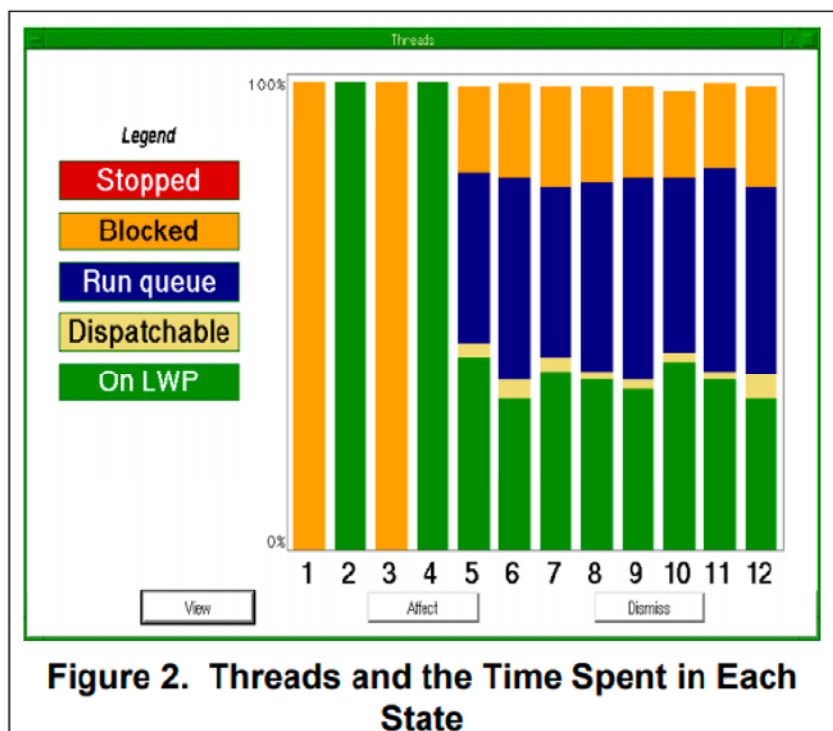
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify IBM xSeries Server Software Rejuvenation Agent to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

902. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. See IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *and performing a corrective action to fix any detected abnormalities,*

903. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Turek ’070. *See* Ex. B-1 at limitation 1.5. A person of ordinary skill in the art would understand that an agent or its runtime environment requesting a corrective policy from an entity outside of the runtime environment would be useful in improving the efficiency of agents in a distributed network

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

904. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

905. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

906. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent IBM xSeries Server Software Rejuvenation Agent does not by itself anticipate this claim element, IBM xSeries Server Software Rejuvenation Agent in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 1; Ex. B-12, claim 1; Turek Anticipation Claim 1.

5. ***Claim 2***

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

907. To the extent IBM xSeries Server Software Rejuvenation Agent does not by itself anticipate this claim element, IBM xSeries Server Software Rejuvenation Agent in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-12, claim 2.

908. IBM xSeries Server Software Rejuvenation Agent discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

909. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as IBM xSeries Server Software Rejuvenation Agent does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.*** The main advantages of threads are:

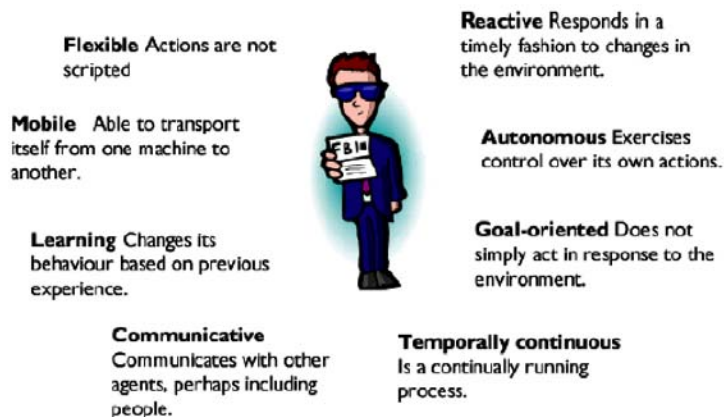
- o ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as ***each thread can be tested independently of other threads.***
- o They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. ***Agent mobility overcomes this by minimizing bandwidth consumption, as they support:***

- ***Adaptive network load balancing.***
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1** Agent properties

Buchanan at Fig. 1.

910. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

911. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* IBM xSeries Server Software Rejuvenation Agent Combination Claim 1.

912. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify IBM xSeries Server Software Rejuvenation Agent to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

6. **Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operational parameters to known thresholds.*

913. IBM xSeries Server Software Rejuvenation Agent discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 3.

7. ***Claim 6***

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

914. IBM xSeries Server Software Rejuvenation Agent discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 6.

915. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

916. To the extent IBM xSeries Server Software Rejuvenation Agent does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, IBM xSeries Server Software Rejuvenation Agent in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* IBM xSeries Server Software Rejuvenation Agent Anticipation Claims 1, 6; IBM xSeries Server Software Rejuvenation Agent Combination Claim 1; Ex. B-12, claim 6.

917. IBM xSeries Server Software Rejuvenation Agent discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

918. To the extent this limitation is not expressly disclosed by IBM xSeries Server Software Rejuvenation Agent, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '659 Patent. IBM xSeries Server Software Rejuvenation Agent discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

919. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in IBM xSeries Server Software Rejuvenation Agent's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

920. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify IBM xSeries Server Software Rejuvenation Agent to create a corrective policy using Dijkstra's Self Stabilization Algorithm in IBM xSeries Server Software Rejuvenation Agent's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. *Claim 7*

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

921. IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

922. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

923. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7. To the extent IBM xSeries Server Software Rejuvenation Agent does not by itself anticipate this claim element, IBM xSeries Server Software Rejuvenation Agent in combination with Cantrill and/or Berry discloses this claim element. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7; *see* IBM xSeries Server Software Rejuvenation Agent Combination Claim 1; Ex. B-12, claim 7.

- d. *detecting if an abnormality exists based on the monitored operational parameters;*

924. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *performing a corrective action to fix any detected abnormalities;*

925. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7.*

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

926. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7.*

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

927. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 7.*

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

928. IBM xSeries Server Software Rejuvenation Agent discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

operation,” for at least the reasons explained above regarding claim 2. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 8. To the extent IBM xSeries Server Software Rejuvenation Agent does not by itself anticipate this claim element, IBM xSeries Server Software Rejuvenation Agent in combination with Buchanan and/or Cantrill discloses this claim element. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 8; *see* IBM xSeries Server Software Rejuvenation Agent Combination Claim 2; Ex. B-12, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

929. IBM xSeries Server Software Rejuvenation Agent discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

930. IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim 13, “[a] system, comprising . . .” ’659 Pat. at Cl. 13. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

- b. *a processor;*

931. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

- c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

932. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

933. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

934. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

935. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

g. *and performing a corrective action to fix any detected abnormalities;*

936. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

937. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

938. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

939. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

940. IBM xSeries Server Software Rejuvenation Agent discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* IBM xSeries Server Software Rejuvenation

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Agent Anticipation Claim 14. To the extent IBM xSeries Server Software Rejuvenation Agent does not by itself anticipate this claim element, IBM xSeries Server Software Rejuvenation Agent in combination with Buchanan and/or Cantrill discloses this claim element. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 14; *see* IBM xSeries Server Software Rejuvenation Agent Combination Claim 2; Ex. B-12, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

941. IBM xSeries Server Software Rejuvenation Agent discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

942. IBM xSeries Server Software Rejuvenation Agent discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent IBM xSeries Server Software Rejuvenation Agent does not by itself anticipate this claim element, IBM xSeries Server Software Rejuvenation Agent in combination with RFC 2328 and/or Afek discloses this claim element. *See* unning times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. ***When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.*** Once the IBM xSeries Server Software Rejuvenation Agent Anticipation Claim 18; *see* Da Rocha Combination Claim 6; Ex. B-12, claim 18.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****T. INCA in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek****1. *Motivation to Combine INCA with Buchanan, Cantrill, and/or Berry***

943. INCA in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. INCA discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of INCA and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in INCA because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and INCA are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or INCA to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. *Motivation to Combine INCA with RFC 2328 and/or Afek***

944. INCA in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. INCA discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of INCA and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in INCA because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and INCA are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or INCA to include these elements, rendering them obvious.

**3. *Motivation to Combine INCA with Turek***

945. INCA in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both INCA and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of INCA and Turek. For example, a person of ordinary skill in

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the art would have been motivated to implement Turek's multithreading in INCA because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in INCA in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that INCA and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify INCA and/or Turek to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A computer-implemented method, comprising:*

946. INCA discloses this claim element. *See* INCA Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

947. INCA discloses this claim element. '659 Pat. at Cl. 1. *See* INCA Anticipation Claim 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. B-13, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

948. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. B-13, claim 1.

949. INCA discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* INCA Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

950. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

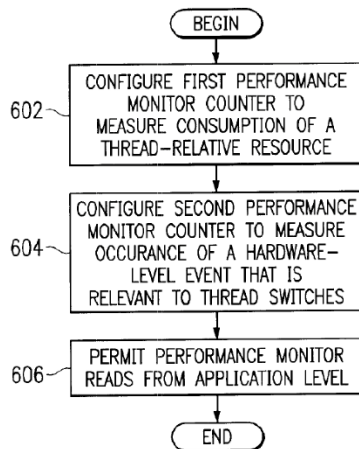
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 6A

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with INCA, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

951. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program's resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

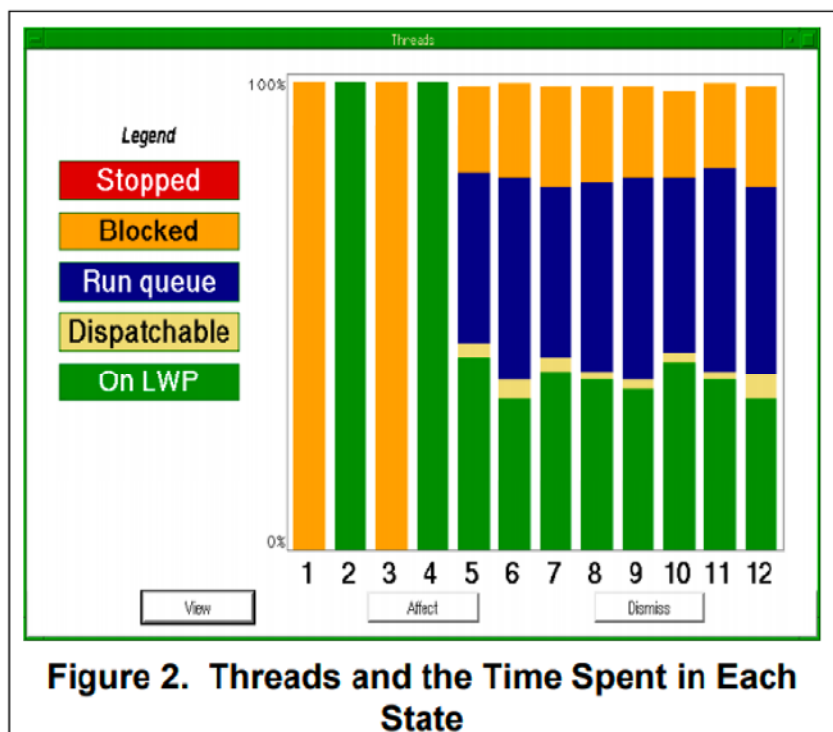
Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors-any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult-it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify INCA to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

952. INCA discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* INCA Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

953. INCA discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* INCA Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

954. INCA discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* INCA Anticipation Claim 1.

g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

955. INCA discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* INCA Anticipation Claim 1.

h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

956. INCA discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. B-13, claim 1; Turek Anticipation Claim 1.

**5. Claim 2**

a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

957. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Pat. at Cl. 2; *see* Turek Anticipation Claim 2; Ex. B-13, claim 2.

958. INCA discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* INCA Anticipation Claims 1, 2. It would have been

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

959. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as INCA does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections.* There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. *It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.* To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract. Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

*Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).* A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). *These threads allow for smoother operation.* A server application that could only handle a request from one client would be of limited use. *Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.* The main advantages of threads are:

- o *They make better use of the processor, where different threads can be run when one or more threads are waiting for data.* For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- o They are easier to test, as *each thread can be tested independently of other threads.*
- o They can use standard threads, which are optimised for given hardware.

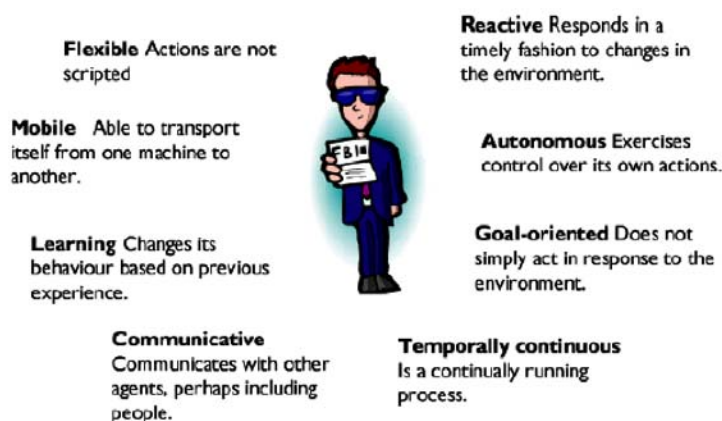
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. *Agent mobility overcomes this by minimizing bandwidth consumption, as they support:*

- *Adaptive network load balancing.*
- Solve problems caused by intermittent or unreliable network connections.

Buchanan at sec. 1.



**Figure 1** Agent properties

Buchanan at Fig. 1.

960. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

961. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* INCA Combination Claim 1.

962. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify INCA

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

963. INCA discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* INCA Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

964. INCA discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* INCA Anticipation Claim 6.

965. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

966. To the extent INCA does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, INCA in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* INCA Anticipation Claims 1, 6; INCA Combination Claim 1; Ex. B-13, claim 6.

967. INCA discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* INCA Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

968. To the extent this limitation is not expressly disclosed by INCA, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. See NetFuel's Infringement Contentions for '659 Patent. INCA discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

969. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in INCA's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

970. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify INCA to create a corrective policy using Dijkstra's Self Stabilization Algorithm in INCA's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the system to perform a method, comprising:*

971. INCA discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* INCA Anticipation Claim 7.

b. *running at least one thread in a first runtime environment;*

972. INCA discloses this element of claim 7, “running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* INCA Anticipation Claim 7.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

973. INCA discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* INCA Anticipation Claim 7. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Cantrill and/or Berry discloses this claim element. *See* INCA Anticipation Claim 7; *see* INCA Combination Claim 1; Ex. B-13, claim 7.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

974. INCA discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* INCA Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

975. INCA discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* INCA Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*first runtime environment,*

976. INCA discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* INCA Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

977. INCA discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* INCA Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

978. INCA discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* INCA Anticipation Claim 8. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Buchanan and/or Cantrill discloses this claim element. *See* INCA Anticipation Claim 8; *see* INCA Combination Claim 2; Ex. B-13, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

979. INCA discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

known thresholds,” for at least the reasons explained above regarding claim 3. *See* INCA Anticipation Claim 9.

11. ***Claim 13***

a. *A system, comprising:*

980. INCA discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* INCA Anticipation Claim 13.

b. *a processor;*

981. INCA discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* INCA Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

982. INCA discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* INCA Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

983. INCA discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* INCA Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

984. INCA discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* INCA Anticipation Claim 13.

f. *detecting there is an abnormality in the monitored operational parameters;*

985. INCA discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* INCA Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- g. *and performing a corrective action to fix any detected abnormalities;*

986. INCA discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* INCA Anticipation Claim 13.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

987. INCA discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* INCA Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

988. INCA discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* INCA Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

989. INCA discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

990. INCA discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* INCA Anticipation Claim 14. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Buchanan and/or Cantrill discloses this claim element. *See* INCA Anticipation Claim 14; *see* INCA Combination Claim 2; Ex. B-13, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

991. INCA discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* INCA Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

992. INCA discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent INCA does not by itself anticipate this claim element, INCA in combination with RFC 2328 and/or Afek discloses this claim element. *See* INCA Anticipation Claim 18; *see* INCA Combination Claim 6; Ex. B-13, claim 18.

**U. Yoshihara in combination with Cantrill/Berry and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine Yoshihara with Cantrill and/or Berry***

993. Yoshihara in combination with Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

distributed networks using software-agent-based scalable solutions. Yoshihara discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Yoshihara and Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Yoshihara because the Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Cantrill and/or Berry and Yoshihara are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Cantrill and/or Berry and/or Yoshihara to include these elements, rendering them obvious.

2. ***Motivation to Combine Yoshihara with RFC 2328 and/or Afek***

994. Yoshihara in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Yoshihara discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management.. The common subject matter and similar approaches to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Yoshihara and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Yoshihara because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Yoshihara are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Yoshihara to include these elements, rendering them obvious.

3. ***Motivation to Combine Yoshihara with Turek***

995. Yoshihara in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Yoshihara and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Yoshihara and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Yoshihara because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement Turek's event monitoring system in Yoshihara in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Yoshihara and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

well known to one of ordinary skill in the art, and it would have been obvious to modify Yoshihara and/or Turek to include these elements, rendering them obvious.

4. *Claim 1*

a. *A computer-implemented method, comprising:*

996. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

997. Yoshihara discloses this claim element. '659 Pat. at Cl. 1. *See* Yoshihara Anticipation Claim 1. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Turek discloses this claim element. '659 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. B-21, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

998. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Cantrill and/or Berry discloses this claim element. '659 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. B-21, claim 1.

999. Yoshihara discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Yoshihara Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

1000. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” '659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed

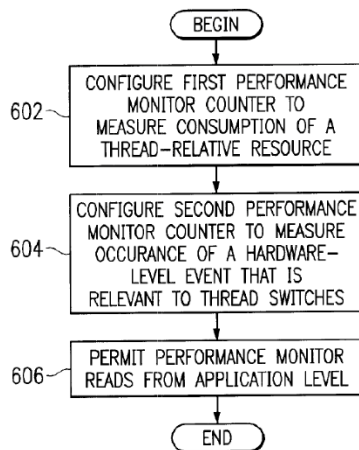
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64.

Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:



*FIG. 6A*

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Yoshihara, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

1001. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors—any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

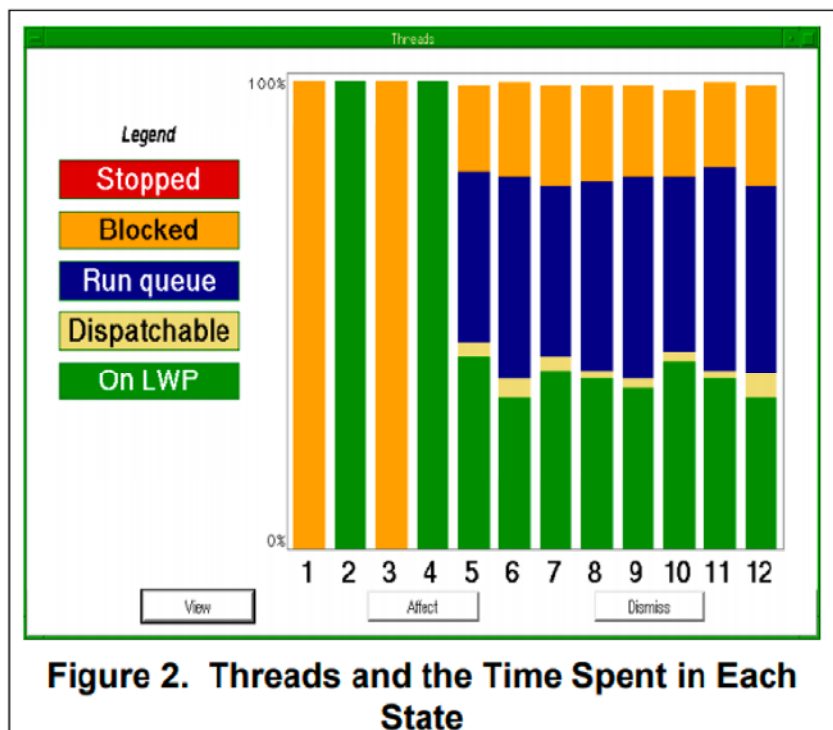
Debugging the scheduler with conventional tools was difficult—it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced—this was clearly*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Yoshihara to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

1002. Yoshihara discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See Yoshihara Anticipation Claim 1.*

- e. *and performing a corrective action to fix any detected abnormalities,*

1003. Yoshihara discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See Yoshihara Anticipation Claim 1.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1004. Yoshihara discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Yoshihara Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1005. Yoshihara discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Yoshihara Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1006. Yoshihara discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Turek discloses this claim element. ’659 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. B-21, claim 1; Turek Anticipation Claim 1.

**5. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*corrective action comprises a load balancing operation.*

1007. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Cantrill discloses this claim element. '659 Pat. at Cl. 2; *see* Yoshihara Anticipation Claim 2; Ex. B-21, claim 2.

1008. Yoshihara discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Yoshihara Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Cantrill.

1009. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” '659 Pat., Cl. 1; *see also* Yoshihara Combination Claim 1.

1010. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Yoshihara to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Cantrill.

**6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1011. Yoshihara discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” '659 Pat. at Cl. 3. *See* Yoshihara Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra's Self Stabilization*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Algorithm.*

1012. Yoshihara discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Yoshihara Anticipation Claim 6.

1013. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

1014. To the extent Yoshihara does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Yoshihara in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Pat. at Cl. 1; *see* Yoshihara Anticipation Claims 1, 6; Yoshihara Combination Claim 1; Ex. B-21, claim 6.

1015. Yoshihara discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Yoshihara Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

1016. To the extent this limitation is not expressly disclosed by Yoshihara, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’659 Patent. Yoshihara discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

1017. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Yoshihara’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

system can be placed in an illegal global state, while each process is individually in a legal state. *The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

1018. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Yoshihara to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Yoshihara's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

8. ***Claim 7***

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*

1019. Yoshihara discloses the preamble of claim 7, "[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . ." '659 Pat. at Cl. 7. *See* Yoshihara Anticipation Claim 7.

- b. *running at least one thread in a first runtime environment;*

1020. Yoshihara discloses this element of claim 7, "running at least one thread in a first runtime environment at least for all the reasons explained above regarding claim 1, element b. *See* Yoshihara Anticipation Claim 7.

- c. *monitoring operational parameters relating to the each thread*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*including a per-thread utilization for the each thread;*

1021. Yoshihara discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Yoshihara Anticipation Claim 7. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Cantrill and/or Berry discloses this claim element. *See* Yoshihara Anticipation Claim 7; *see* Yoshihara Combination Claim 1; Ex. B-21, claim 7.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

1022. Yoshihara discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Yoshihara Anticipation Claim 7.

e. *performing a corrective action to fix any detected abnormalities;*

1023. Yoshihara discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Yoshihara Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1024. Yoshihara discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Yoshihara Anticipation Claim 7.

g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*multiple threads.*

1025. Yoshihara discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Yoshihara Anticipation Claim 7.

9. ***Claim 8***

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

1026. Yoshihara discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Yoshihara Anticipation Claim 8. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Cantrill discloses this claim element. *See* Yoshihara Anticipation Claim 8; *see* Yoshihara Combination Claim 2; Ex. B-21, claim 8.

10. ***Claim 9***

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1027. Yoshihara discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Yoshihara Anticipation Claim 9.

11. ***Claim 13***

- a. *A system, comprising:*

1028. Yoshihara discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Yoshihara Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

b. *a processor;*

1029. Yoshihara discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See Yoshihara Anticipation Claim 13.*

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1030. Yoshihara discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See Yoshihara Anticipation Claim 13.*

d. *running at least one thread in a first runtime environment;*

1031. Yoshihara discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Yoshihara Anticipation Claim 13.*

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1032. Yoshihara discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See Yoshihara Anticipation Claim 13.*

f. *detecting there is an abnormality in the monitored operational parameters;*

1033. Yoshihara discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See Yoshihara Anticipation Claim 13.*

g. *and performing a corrective action to fix any detected abnormalities;*

1034. Yoshihara discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See Yoshihara Anticipation Claim 13.*

h. *wherein performing the corrective action comprises first making a*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1035. Yoshihara discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Yoshihara Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1036. Yoshihara discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Yoshihara Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1037. Yoshihara discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

12. ***Claim 14***

- a. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1038. Yoshihara discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Yoshihara Anticipation Claim 14. To the extent Yoshihara does not by itself

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

anticipate this claim element, Yoshihara in combination with Cantrill discloses this claim element. See Yoshihara Anticipation Claim 14; see Yoshihara Combination Claim 2; Ex. B-21, claim 14.

13. ***Claim 15***

- a. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1039. Yoshihara discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. See Yoshihara Anticipation Claim 15.

14. ***Claim 18***

- a. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1040. Yoshihara discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with RFC 2328 and/or Afek discloses this claim element. See Yoshihara Anticipation Claim 18; see Yoshihara Combination Claim 6; Ex. B-21, claim 18.

**V. Cisco Catalyst 5000 Switch Family in combination with NetRanger and Goldman**

1. ***Motivation to Combine Cisco Catalyst 5000 Switch with NetRanger and/or Goldman***

1041. It would have been obvious to combine the Cisco Catalyst 5000 Switch Family with NetRanger, because the combination is merely the use of a known technique to improve a similar device, method, or product in the same way. NetRanger discloses automatically configuring security policies in response to detected events by reconfiguring the Access Control Lists (ACLs) of a Cisco switch, and the Catalyst 5000 Switch family contains ACLs for security and QoS matching. Cisco Catalyst 5000 Switch Family in combination with NetRanger and/or Goldman

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses and/or renders obvious all elements of the '730 Patent. Ex. A-9. Cisco Catalyst 5000 Switch Family, NetRanger, and Goldman are all in the field of software network management.

2. ***Motivation to Combine Cisco NetRanger with Buchanan, Cantrill, and/or Berry***

1042. Cisco NetRanger in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco NetRanger discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco NetRanger and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Cisco NetRanger because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial.. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Cisco NetRanger are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

been obvious to modify Buchanan, Cantrill, and/or Berry and/or Cisco NetRanger to include these elements, rendering them obvious.

**3. *Motivation to Combine Cisco NetRanger with RFC 2328 and/or Afek***

1043. Cisco NetRanger in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco NetRanger discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco NetRanger and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Cisco NetRanger because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Cisco NetRanger are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Cisco NetRanger to include these elements, rendering them obvious.

**W. *Lindskog in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek and/or Turek***

**1. *Motivation to Combine Lindskog with Buchanan, Cantrill, and/or Berry***

1044. Lindskog in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

subject matter, management of distributed networks using software-agent-based scalable solutions. Lindskog discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Lindskog and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Lindskog because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Lindskog are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Lindskog to include these elements, rendering them obvious.

***2. Motivation to Combine Lindskog with RFC 2328 and/or Afek***

1045. Lindskog in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Lindskog discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Lindskog and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Lindskog because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Lindskog are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Lindskog to include these elements, rendering them obvious.

**3. *Motivation to Combine Lindskog with Turek***

1046. Lindskog in combination with Turek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Both Lindskog and Turek disclose using mobile software agents to monitor and correct faults in a distributed network. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Lindskog and Turek. For example, a person of ordinary skill in the art would have been motivated to implement Turek's multithreading in Lindskog because multithreading would have improved the efficiency of agents in a distributed network, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

art would have been motivated to implement Turek's event monitoring system in Lindskog in order to achieve the predictable result of improving the efficiency of a distributed networking system through an entity monitoring events in multiple runtime environments/across multiple agents. Additionally, to the extent that Lindskog and Turek are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Lindskog and/or Turek to include these elements, rendering them obvious.

**4. Claim 1**

a. *A computer-implemented method, comprising:*

1047. Lindskog discloses this claim element. *See* Lindskog Anticipation Claim 1.

b. *running at least one thread in a first runtime environment;*

1048. Lindskog discloses this claim element. '659 Patent claim 1. *See* Lindskog Anticipation Claim 1. To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Turek discloses this claim element. '659 Patent claim 1; *see* Lindskog Anticipation Claim 1; Ex. B-18, claim 1; Turek Anticipation Claim 1.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1049. To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Cantrill and/or Berry discloses this claim element. '659 Patent claim 1; *see* Lindskog Anticipation Claim 1; Ex. B-18, claim 1.

1050. Lindskog discloses agents having associated underlying resources and monitoring events and parameters associated with the underlying resources. *See* Lindskog Anticipation Claim 1. It would have been obvious at the time of invention to a person of ordinary skill in the art to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1051. Berry discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Berry discloses a method and apparatus “for low-overhead performance measurement of an application executing in a data processing system in order to generate per thread performance information in a multithreaded environment.” Berry at 3:42-45. Berry discloses monitoring a first “thread-relative” metric and a second metric relating to “events that may indirectly cause inaccuracies in the first metric.” Berry at 3:42-64. “For example, the first metric could be a value of a consumed resource, such as a number of executed instructions, while the second metric is a number of interrupts, each of which might cause the kernel to initiate a thread switch.” Berry at 3:42-64. Further, Berry discloses:

With reference now to FIG. 6A, a flowchart depicts the steps that may be necessary to configure a processor for counting events in accordance with a first embodiment of the present invention. *A first performance monitor counter is configured to measure the consumption of a thread-relative resource, such as counting the number of executed instructions (step 602). A second performance monitor counter is configured to measure occurrence of a hardware-level event that is relevant to thread switches, such as interrupts (step 604). For example, a monitor mode control register in the processor can be configured to control the performance monitor counters by executing one or more specialized instructions for this purpose.*

Berry at 12:65-13:10. Fig. 6 of Berry further discloses the per-thread utilization monitoring of Berry’s network performance monitoring system:

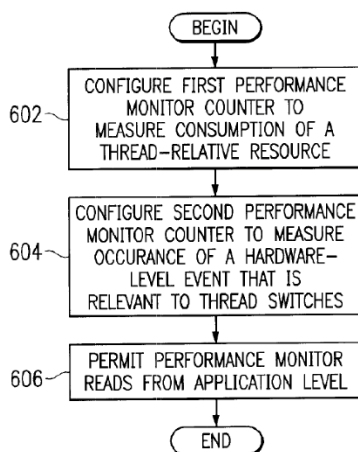


FIG. 6A



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Berry at Fig. 6A. Berry provides express motivation to combine, explaining that “[p]erformance monitoring is often used to optimize the use of software in a system.” Berry at 2:5-6. Berry provides further explicit motivation to combine with Lindskog, explaining the benefits of its thread-level performance monitoring system, which a person of ordinary skill in the art would have recognized as being beneficial:

*Effective management and enhancement of data processing systems requires knowing how and when various system resources are being used.* In analyzing and enhancing performance of a data processing system and the applications executing within the data processing system, it is helpful to know which software modules within a data processing system are using system resources. *Performance tools are used to monitor and examine a data processing system to determine resource consumption as various software applications are executing within the data processing system. For example, a performance tool may identify the most frequently executed modules and instructions in a data processing system, may identify those modules which allocate the largest amount of memory, or may identify those modules which perform the most I/O requests.* Hardware-based performance tools may be built into the system and, in some cases, may be installed at a later time, while software-based performance tools may generally be added to a data processing system at any time.

Berry at 1:18-36.

1052. Cantrill also discloses “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” ’659 Pat., Cl. 1. For example, Cantrill discloses a ThreadMon tool for thread-level resource management and monitoring:

1) Bottleneck analysis: concurrent programs consist of a number of threads, each executing instructions independently and competing for various resources. Contention for these resources hinders performance — thus its minimization is an important goal. *By interposing itself between the application and the threads package, ThreadMon can monitor a program’s resource usage and display the extent of contention, not only for individual resources but for aggregates of resources.* Compounding this resource-contention problem is that many library routines cause contention for resources that the application programmer may not even know exist. *Our tool identifies and shows the conflicts for these resources, providing further valuable information to the programmer.*

Cantrill at p. 253, sec. 1.

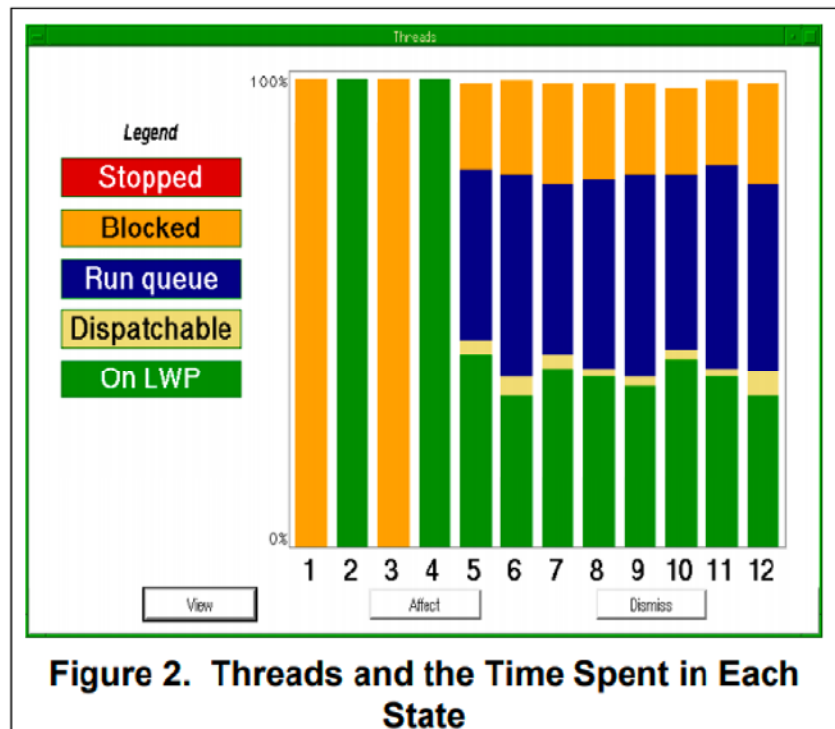
The program is driven by the desired frame rate. The more time available to compute, the more detailed is the frame produced. The time required for

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

each of the compute and rendering tasks is known; the scheduler's job is to ensure that enough computation tasks are performed that sufficient detail is available for each frame, but that these tasks are done in the time allotted to produce a frame. *To obtain the best use of a multiprocessor, the scheduler should distribute the computing tasks evenly over all of the processors-any idle time on a processor is time that could have been spent performing a task assigned to another processor (and thus speeding up the computation) or performing an additional task (and thus adding more detail to the frame).*

Debugging the scheduler with conventional tools was difficult-it was not easy, for example, to verify that its determination of the running times of the various tasks was correct and that the schedule produced resulted in the balanced use of the processors. *When we first applied ThreadMon to the program we found that there were unsuspected bugs in the scheduler and that the processor usage was, indeed, unbalanced-this was clearly indicated by both the activity displays of the compute threads and the activity displays of the synchronization variables.* Once the problem was identified, it was easily fixed and the solution verified by monitoring the program with ThreadMon.

Cantrill at p. 260, sec. 5.2.



Cantrill at Fig. 2. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Lindskog to monitor operational parameters, including per-thread utilization, for each thread or resource in a distributed networking system, as disclosed by Cantrill and/or Berry.

- d. *detecting if there is an abnormality in the monitored operational parameters;*

1053. Lindskog discloses this element of claim 1, “detecting if there is an abnormality in the monitored operational parameters.” ’659 Pat. at Cl. 1. *See* Lindskog Anticipation Claim 1.

- e. *and performing a corrective action to fix any detected abnormalities,*

1054. Lindskog discloses this element of claim 1, “and performing a corrective action to fix any detected abnormalities.” ’659 Pat. at Cl. 1. *See* Lindskog Anticipation Claim 1.

- f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1055. Lindskog discloses this element of claim 1, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Lindskog Anticipation Claim 1.

- g. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1056. Lindskog discloses this element of claim 1, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” ’659 Pat. at Cl. 1. *See* Lindskog Anticipation Claim 1.

- h. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1057. Linskog discloses this element of claim 1, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.” ’659 Pat. at Cl. 1. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Turek discloses this claim element. ’659 Patent claim 1; *see* Linskog Anticipation Claim 1; Ex. B-18, claim 1; Turek Anticipation Claim 1.

**5. Claim 2**

- a. *The computer-implemented method of claim 1, wherein the corrective action comprises a load balancing operation.*

1058. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Cantrill and/or Buchanan discloses this claim element. ’659 Patent claim 2; *see* Linskog Anticipation Claim 2; Ex. B-18, claim 2.

1059. Linskog discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Linskog Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to perform corrective actions, including load-balancing operations by agents in a distributed network in order to achieve improved network efficiency, as disclosed by Buchanan and/or Cantrill.

1060. Buchanan discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1. For example, Buchanan discloses an adaptive load balancing using a system of mobile agents that utilizes a Java framework, just as Linskog does:

*Agent mobility addresses some limitations faced by classic client/server architecture, namely, in minimising bandwidth consumption, in supporting adaptive network load balancing and in solving problems caused by intermittent or unreliable network connections. There has been a great deal of attention on the potential productivity gains expected from so-called intelligent agents. These however require complex artificial intelligence (AI) functionality. Agents can realistically be of benefit in those areas concerned with autonomy and mobility. This is especially true of*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

network management applications and this will be the focus of this paper. The paper discusses the usage of mobile agents and the advantages that these have over traditional client/server applications. ***It discusses the main characteristics of an agent, and shows how Java has the main components that allow mobile agents to be easily development.*** To show how agents are implemented it gives a practical implemented of an agent. Finally, the paper also discusses the main Java agent development systems, which are IBM aglets, Object Space Voyager and JATLite and outlines the advantages of using each of them.

Buchanan at Abstract (emphasis added). Buchanan further discloses the benefits associated with utilizing multiple threads to perform multiple concurrent tasks in order to improve system efficiency:

***Threads. Multitasking involves running several processes at a time. Multitasking programs split into a number of parts (threads) and each of these is run on the multitasking system (multithreading).*** A program which is running more than one thread at a time is known as a multithreaded program (Figure 5). ***These threads allow for smoother operation.*** A server application that could only handle a request from one client would be of limited use. ***Threads provide a means to allow an application to perform multiple tasks simultaneously. Java makes creating, controlling, and co-ordinating threads relatively simple.*** The main advantages of threads are:

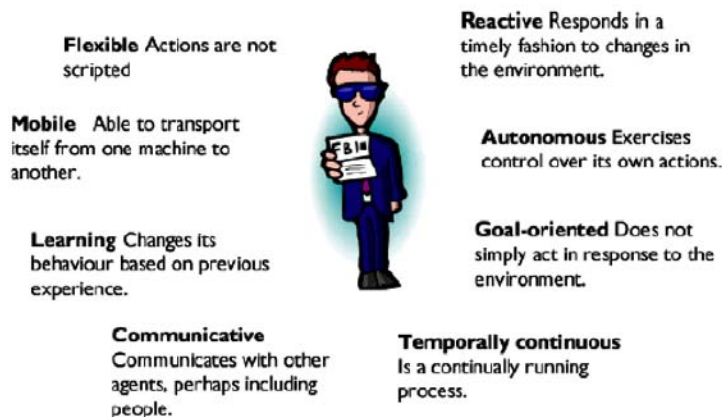
- ***They make better use of the processor, where different threads can be run when one or more threads are waiting for data.*** For example, a thread could be waiting for keyboard input, while another thread could be reading data from the disk.
- They are easier to test, as ***each thread can be tested independently of other threads.***
- They can use standard threads, which are optimised for given hardware.

Buchanan at sec. 6. Buchanan further discloses that its mobile agents support “network load balancing,” which a person of ordinary skill in the art would have recognized as beneficial due to its potential for minimizes bandwidth consumption and associated performance benefits:

Traditional client/server architectures are typically wasteful in their usage of bandwidth. ***Agent mobility overcomes this by minimizing bandwidth consumption, as they support:***

- Adaptive network load balancing.
- Solve problems caused by intermittent or unreliable network connections.”

Buchanan at sec. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1** Agent properties

Buchanan at Fig. 1.

1061. Buchanan also provides further express motivation for the combination, acknowledging that “[t]here are other advantages to be gained including real time notification where an agent situated at some remote site may notify a local host of any important event immediately. *Also, parallel execution (or load balancing) where a large computation can be divided dependent on resources. All these offer compelling reasons to adopt agent architecture for network management tasks.*” Buchanan at sec. 4.

1062. As described above, Cantrill also discloses “wherein the corrective action comprises a load balancing operation.” ’659 Pat., Cl. 1; *see also* Lindskog Combination Claim 1.

1063. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Lindskog to perform a corrective action that comprised a load-balancing operation in its multi-threaded distributed networking management system, as disclosed by Buchanan and/or Cantrill.

### **6. Claim 3**

- a. *The computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1064. Lindskog discloses claim 3, “[t]he computer-implemented method of claim 1, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.” ’659 Pat. at Cl. 3. *See* Lindskog Anticipation Claim 3.

**7. Claim 6**

- a. *The computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1065. Lindskog discloses claim 6, “[t]he computer-implemented method of claim 1, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.” ’659 Pat. at Cl. 6. *See* Lindskog Anticipation Claim 6.

1066. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra’s algorithm in the model.

1067. To the extent Lindskog does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Lindskog in combination with RFC 2328 and/or Afek discloses this claim element. ’659 Patent claim 1; *see* Lindskog Anticipation Claims 1, 6; Lindskog Combination Claim 1; Ex. B-18, claim 6.

1068. Lindskog discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Lindskog Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

1069. To the extent this limitation is not expressly disclosed by Lindskog, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

See NetFuel's Infringement Contentions for '659 Patent. Lindskog discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

1070. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Lindskog's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at p. 200, Sec. 1.

1071. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Lindskog to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Lindskog's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

#### **8. Claim 7**

- a. *A non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising:*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1072. Linskog discloses the preamble of claim 7, “[a] non-transitory computer-readable medium comprising a sequence of instructions which when executed by a system causes the system to perform a method, comprising . . . .” ’659 Pat. at Cl. 7. *See* Linskog Anticipation Claim 7.

b. *running at least one thread in a first runtime environment;*

1073. Linskog discloses this element of claim 7, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Linskog Anticipation Claim 7.

c. *monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread;*

1074. Linskog discloses this element of claim 7, “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Linskog Anticipation Claim 7. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Cantrill and/or Berry discloses this claim element. *See* Linskog Anticipation Claim 7; *see* Linskog Combination Claim 1; Ex. B-18, claim 7.

d. *detecting if an abnormality exists based on the monitored operational parameters;*

1075. Linskog discloses this element of claim 7, “detecting if an abnormality exists based on the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Linskog Anticipation Claim 7.

e. *and performing a corrective action to fix any detected abnormalities;*

1076. Linskog discloses this element of claim 7, “performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Linskog Anticipation Claim 7.

f. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1077. Lindskog discloses this element of claim 7, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Lindskog Anticipation Claim 7.

- g. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

1078. Lindskog discloses this element of claim 7, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h. *See* Lindskog Anticipation Claim 7.

**9. Claim 8**

- a. *The computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation.*

1079. Lindskog discloses claim 8, “[t]he computer-readable medium of claim 7, wherein the corrective action comprises a load balancing operation,” for at least the reasons explained above regarding claim 2. *See* Lindskog Anticipation Claim 8. To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Buchanan and/or Cantrill discloses this claim element. *See* Lindskog Anticipation Claim 8; *see* Lindskog Combination Claim 2; Ex. B-18, claim 8.

**10. Claim 9**

- a. *The computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1080. Lindskog discloses claim 9, “[t]he computer-readable medium of claim 7, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” for at least the reasons explained above regarding claim 3. *See* Lindskog Anticipation Claim 9.

**11. Claim 13**

a. *A system, comprising:*

1081. Lindskog discloses the preamble of claim 13, “[a] system, comprising . . . .” ’659 Pat. at Cl. 13. *See* Lindskog Anticipation Claim 13.

b. *a processor;*

1082. Lindskog discloses this element of claim 13, “a processor.” ’659 Pat. at Cl. 13. *See* Lindskog Anticipation Claim 13.

c. *and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising:*

1083. Lindskog discloses this element of claim 13, “and a memory coupled to the processor, the memory storing instructions which when executed by the processor causes the system to perform a method comprising.” ’659 Pat. at Cl. 13. *See* Lindskog Anticipation Claim 13.

d. *running at least one thread in a first runtime environment;*

1084. Lindskog discloses this element of claim 13, “running at least one thread in a first runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Lindskog Anticipation Claim 13.

e. *monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread;*

1085. Lindskog discloses this element of claim 13, “monitoring operational parameters relating to the or each thread including a per-thread utilization for each thread,” at least for all the reasons explained above regarding claim 1, element c. *See* Lindskog Anticipation Claim 13.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *detecting there is an abnormality in the monitored operational parameters;*

1086. Linskog discloses this element of claim 13, “detecting there is an abnormality in the monitored operational parameters,” at least for all the reasons explained above regarding claim 1, element d. *See* Linskog Anticipation Claim 13.

- g. *and performing a corrective action to fix any detected abnormalities;*

1087. Linskog discloses this element of claim 13, “and performing a corrective action to fix any detected abnormalities,” at least for all the reasons explained above regarding claim 1, element e. *See* Linskog Anticipation Claim 13.

- h. *wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,*

1088. Linskog discloses this element of claim 13, “wherein performing the corrective action comprises first making a request for a corrective policy to correct a detected abnormality from an entity external to the first runtime environment if the corrective policy is not available to an agent operating within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element f. *See* Linskog Anticipation Claim 13.

- i. *wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,*

1089. Linskog discloses this element of claim 13, “wherein performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment,” at least for all the reasons explained above regarding claim 1, element g. *See* Linskog Anticipation Claim 13.

- j. *wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1090. Linskog discloses this element of claim 13, “wherein the entity external to the first runtime environment is a global modeler configured to listen to events from a system comprising a plurality of runtime environments including the first runtime environment, each of the runtime environments running multiple threads,” at least for all the reasons explained above regarding claim 1, element h.

**12. Claim 14**

- k. *The system of claim 13, wherein the corrective action comprises a load balancing operation.*

1091. Linskog discloses claim 14, “[t]he system of claim 13, wherein the corrective action comprises a load balancing operation,” at least for all the reasons explained above regarding claim 2. *See* Linskog Anticipation Claim 14. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Buchanan and/or Cantrill discloses this claim element. *See* Linskog Anticipation Claim 14; *see* Linskog Combination Claim 2; Ex. B-18, claim 14.

**13. Claim 15**

- l. *The system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds.*

1092. Linskog discloses claim 15, “[t]he system of claim 13, wherein detecting if an abnormality exists comprises comparing the monitored operational parameters to known thresholds,” at least for all the reasons explained above regarding claim 3. *See* Linskog Anticipation Claim 15.

**14. Claim 18**

- m. *The system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm.*

1093. Linskog discloses claim 18, “[t]he system of claim 13, wherein creating the corrective policy comprises using Dijkstra’s Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with RFC 2328 and/or Afek

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses this claim element. *See* Linskog Anticipation Claim 18; *see* Linskog Combination Claim 6; Ex. B-18, claim 18.

**X. ANTICIPATORY PRIOR ART FOR THE '730 PATENT****A. Anticipation by and/or Obviousness in View of United States Patent No. 6,460,070, filed 6/3/1998 and issued to Turek, et al. on 10/2/2002.**

1094. As explained in detail below and in the chart attached as Ex. A-1, Turek anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

**1. Claim 1**

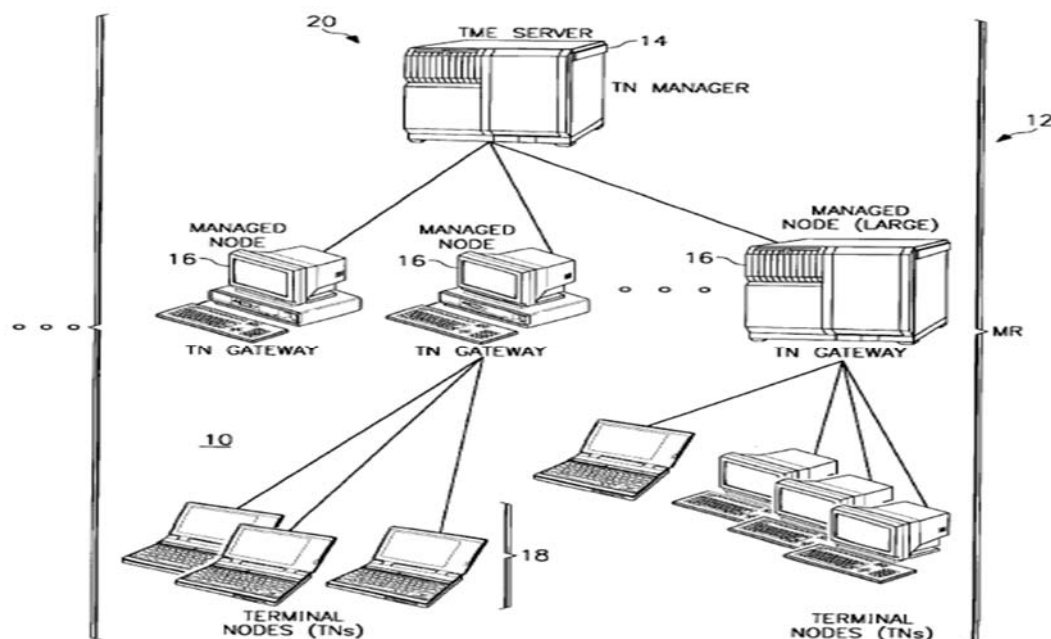
a. *A method of managing a computer network, comprising:*

1095. Turek discloses the preamble of claim 1, "[a] method of managing a computer network, comprising." '730 Pat. at Cl. 1. For example:

The present invention is directed to ***managing a large distributed computer enterprise environment*** and, more particularly, to diagnosing and correcting network faults in such an environment using mobile software agents.

Turek at 1:7-10.

1096. Further examples include Turek at 2:26-32, 3:47-57, 6:62-7:4, Fig. 1.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1097. Turek discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

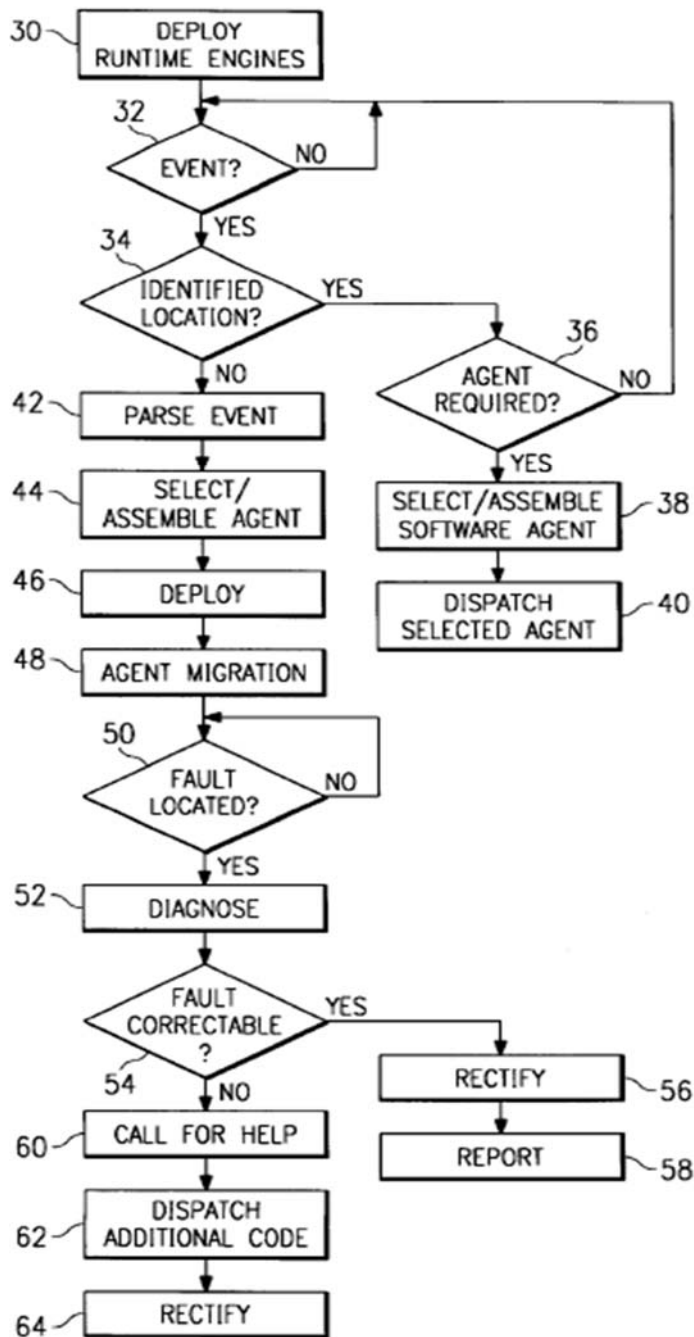
It is another primary *object* of this invention to deploy a *software "agent"* into a distributed computer network environment *to diagnose and, if possible, correct a fault.*

Turek at 2:1-3.

1098. Further examples include Turek at 2:15-20 (“Another object of this invention is to deploy a self-routing software agent into a distributed computer network to locate and correct a network fault or to address some other network event. Preferably, the software agent is a minimum

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

set of tasks that are identified for use in diagnosing and/or correcting the fault.”); Turek at 2:47-53, 2:63-64, 6:6-9, 6:49-59, 7:49-57, 9:11-16, 9:21-29, 7:11-15, 7:30-41, Fig. 4, Fig. 5.



*FIG. 4*



## HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY

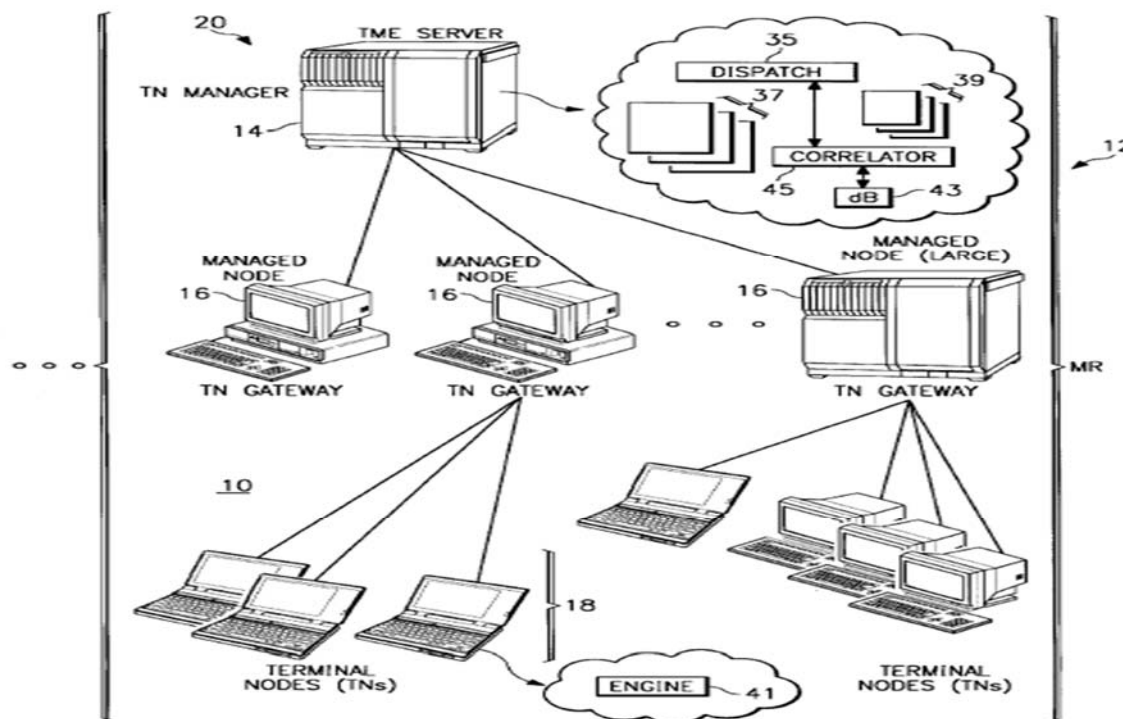


FIG. 5

- c. *is able to communicate with other software agents in the computer network;*

1099. Turek discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

If, however, the event originated elsewhere, the software agent identifies a subset of nodes in the computer network that remain possible candidates for origination of the event. Preferably, the software agent then replicates itself to create a new instance. This new instance is then launched to the identified subset to continue searching for the location and cause of the event. ***At each node, this process is repeated until the location is identified.***

Turek at 2:55-62.

1100. Further examples include Turek at 2:66-3:3 (“As noted above, when the event is a fault, the software agent locates the fault and attempts to rectify it. If necessary, ***the software agent may obtain additional code from the dispatch mechanism or some other network source. Such additional code may be another software agent***”); Turek at 8:1-13, 9:21-29, Fig. 4.

1101. To the extent this limitation is not disclosed by Turek, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *is capable of perceiving its own state*

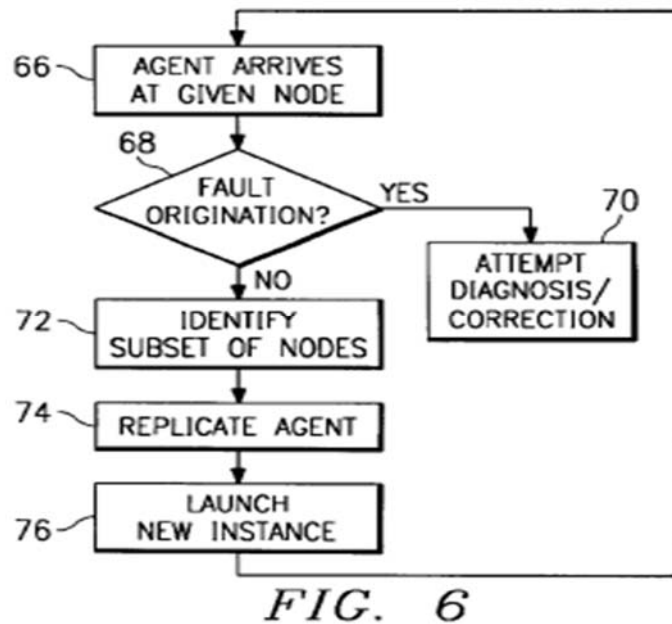
1102. Turek discloses this element of claim 1, “is capable of perceiving its own state.”

’730 Pat. at Cl. 1. For example:

When the software agent is received at a given node, the method determines whether the event originated from the node. If so, the software agent identifies the cause and, if possible, undertakes a corrective or other action depending on the nature of the event in question. Thus, for example, if the event were a fault, the software agent attempts to correct the fault. If, however, the event originated elsewhere, the software agent identifies a subset of nodes in the computer network that remain possible candidates for origination of the event. Preferably, ***the software agent then replicates itself to create a new instance***. This new instance is then launched to the identified subset to continue searching for the location and cause of the event. At each node, this process is repeated until the location is identified.

Turek at 2:49-62.

1103. Further examples include Turek at 2:63-64, 2:66-3:03, 6:49-59, 5:43-53, 7:58-65, 6:66-7:14, 9:21-29, Fig. 4, Fig. 6.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *and is able to clone itself,*

1104. Turek discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at

Cl. 1. For example:

If, however, the event originated elsewhere, the software agent identifies a subset of nodes in the computer network that remain possible candidates for origination of the event. ***Preferably, the software agent then replicates itself to create a new instance. This new instance is then launched to the identified subset to continue searching for the location and cause of the event.*** At each node, this process is repeated until the location is identified.

Turek at 2:55-63.

1105. Further examples include Turek at 6:60-7:2, Fig. 4, Fig. 6.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1106. Turek discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

At step 44, the dispatch mechanism ***selects an appropriate software agent based on the event and/or information retrieved from the database and the event correlator. As used herein, the selection process of step 44 may involve compiling one or more software tasks into a "custom" software agent for this purpose.*** Thus, the present invention covers the use of an existing software agent, as well as an agent that is created or generated "on-the-fly" as a result of a given network event that requires diagnosis and/or correction.

Turek at 7:49-57.

1107. Further examples include Turek at 2:01-03 (“It is another primary object of this invention to deploy a software ‘agent’ into a distributed computer network environment ***to diagnose and, if possible, correct a fault.***”); Turek at 2:21-25, 2:37-45, 2:47-55, 5:43-53, 5:65-6:2, 6:49-59, 2:37-46, 6:66-7:15, 7:34-48, Fig. 4, Fig. 5.

g. *monitoring the computer network;*

1108. Turek discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

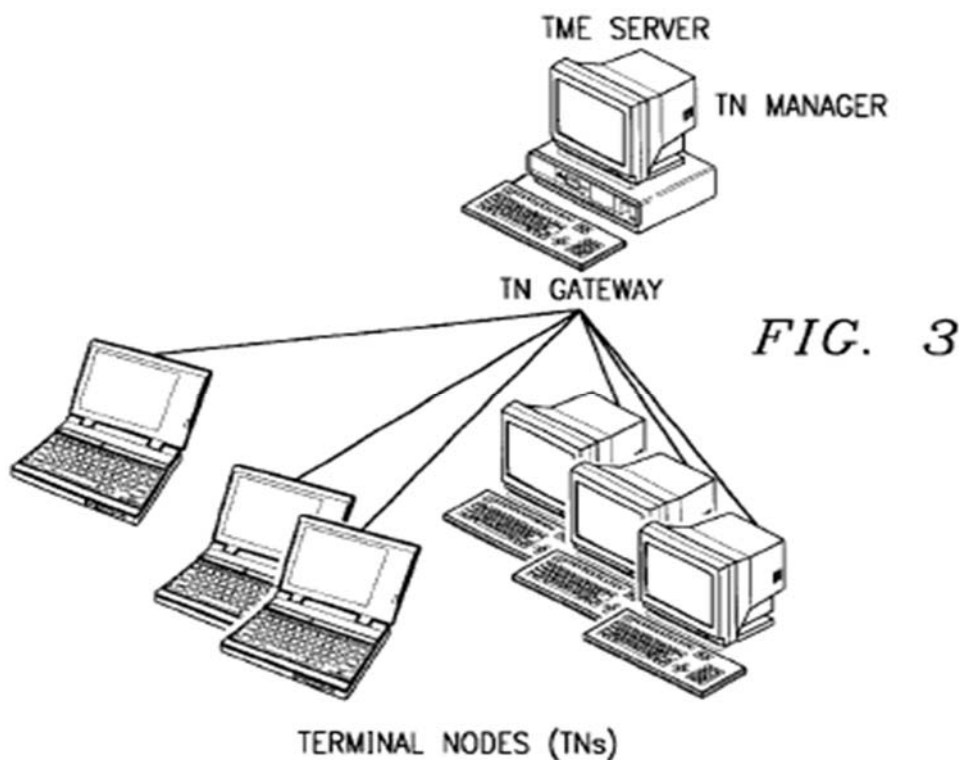
Preferably, the managed environment (ME) is logically broken down into a series of loosely-connected managed regions (MR) 12, each with its own management server 14 for managing local resources with the MR. The network typically will include other servers (not shown) for carrying out other distributed network functions. These include name servers, security

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

servers, file servers, threads servers, time servers and the like. Multiple servers 14 coordinate activities across the enterprise and permit remote site management and operation. Each server 14 serves a number of gateway machines 16, each of which in turn support a plurality of endpoints 18. The server 14 coordinates all activity within the MR using a terminal node manager 20.

Turek at 3:51-64.

1109. Further examples include Turek at 5:43-48 (“***Thus, when a network error or ‘fault’ is reported*** whose cause and location are not apparent or readily ascertainable, an appropriate agent is identified and dispatched to determine this information. Preferably, the agent is dispatched to the actual node in the network at which the fault condition occurs.”); Turek at 3:59-64, 4:22-28, 4:32-41, 4:50-58, 6:66-7:15, 8:44-50, Fig. 3, Fig. 4.



- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the computer network,*

1110. Turek discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

At step 54, a test is done to determine whether the software agent (either alone, or together with some functionality provided by the runtime engine) can correct the problem. If the outcome of the test at step 54 is positive, the routine continues at step 56 and the problem is rectified. ***At step 58, information about the problem and the corrective action that as undertaken are reported back to the dispatch mechanism and stored in the database for future use. If, however, the outcome of the test at step 54 indicates that the software engine and/or the runtime engine cannot rectify the problem, a call is made at step 60 to obtain additional help.*** In a preferred embodiment, dispatch mechanism 62 responds to the call and dispatches additional code (e.g., another software engine) to the node to assist in the problem diagnosis and/or correction, as the case may be. At step 64, the fault is rectified. This completes the processing.

Turek at 8:1-17.

1111. Further examples include Turek at 2:66-3:3 (“As noted above, when the event is a fault, the software agent locates the fault and attempts to rectify it. ***If necessary, the software agent may obtain additional code from the dispatch mechanism or some other network source.*** Such additional code may be another software agent”); Turek at 9:21-29, 7:34-48, 8:44-50, Fig. 4, Fig. 5.

1112. To the extent this limitation is not disclosed by Turek, it is rendered obvious in combination with Kasteleijn or Hellersetin. *See* Ex. A-3, limitation 1.7, Ex. A-20.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1113. Turek discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

At step 54, a test is done to determine whether the software agent (either alone, or together with some functionality provided by the runtime engine) can correct the problem. . . . ***If, however, the outcome of the test at step 54 indicates that the software engine and/or the runtime engine cannot rectify the problem, a call is made at step 60 to obtain additional help.*** In a preferred embodiment, dispatch mechanism 62 responds to the call and dispatches additional code (e.g., another software engine) to the node to assist in the problem diagnosis and/or correction, as the case may be. At step 64, the fault is rectified. This completes the processing.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Turek at 8:1-17.

1114. Further examples include Turek at 2:66-3:3 (“As noted above, when the event is a fault, the software agent locates the fault and attempts to rectify it. *If necessary, the software agent may obtain additional code from the dispatch mechanism or some other network source.* Such additional code may be another software agent”); Turek at 9:21-29, 7:34-48, 8:43-49, 3:51-64, 4:42-49, 4:22-28, Fig. 4, Fig. 5.

1115. To the extent this limitation is not disclosed by Turek, it is rendered obvious in combination with Goldman or Hellersetin. *See* Ex. A-3, limitation 1.8, Ex. A-20.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1116. Turek discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

Once the fault or other problem has been diagnosed, the agent attempts to fix the problem[.] The agent may have the necessary code or it may send requests to the dispatch mechanism for additional code to effect the repair. The to [sic] dispatch mechanism may be located in distributed locations as well, if desired. The additional code may be other software agent(s). If unable to effect repairs, *the agent will, at a minimum, report back with the diagnosis to a user interface of the dispatch mechanism.*

Turek at 9:21-29.

1117. Further examples include Turek at 8:44-50 (“At the same time, information collected by the software agent and/or the runtime engine is returned to the dispatch mechanism and stored in the database for future use. In this way, even though the given node was not the source of the fault, the information obtained during the test at step 70 may be added to the database to enhance system operation in the future.”); Turek at 2:66-3:03, 8:1-17, 7:32-41, 3:51-64, 4:42-49, 4:22-28, Fig. 4, Fig. 5.

1118. To the extent this limitation is not disclosed by Turek, it is rendered obvious in combination with Goldman or Hellersetin. *See* Ex. A-3, limitation 1.9, Ex. A-20.

k. *dynamically modifying the assigned goal of the software agent by*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*replacing the assigned goal based on the optimal policy*

1119. Turek discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl.

1. For example:

***At the same time, information collected by the software agent and/or the runtime engine is returned to the dispatch mechanism and stored in the database for future use.*** In this way, even though the given node was not the source of the fault, ***the information obtained during the test at step 70 may be added to the database to enhance system operation in the future.***

Turek at 8:44-50.

1120. Further examples include Turek at 2:66-3:03, 8:1-17, 9:21-29, 7:34-48, Fig. 4, Fig.

5.

1121. To the extent this limitation is not disclosed by Turek, it is rendered obvious in combination with Goldman or Hellersetin. *See* Ex. A-3, limitation 1.10, Ex. A-20.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1122. Turek discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

Once the management infrastructure is in place, the actual operating routine begins. At step 32, a test is performed at the dispatch mechanism 15 to determine whether a given event has occurred. As noted above, for illustrative purposes, a given event is a network "fault", alarm or other such trigger. One of ordinary skill will appreciate that the particular event need not be a fault or alarm type of condition, however. An alternative event might be a request for maintenance in some non-specific area of the network. The software agent might then be deployed to the general area from which the request originated but then used to identify a specific target location at which the request will be serviced. ***The dispatch mechanism preferably includes an appropriate user interface to enable an administrator to identify event(s). Alternatively, event consumers may subscribe to the dispatch mechanism by identifying particular events of which they are interested in receiving notice.***

Turek at 6:66-7:14.

1123. Further examples include Turek at 2:66-3:03, 8:1-17, 9:21-29, Fig. 4.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1124. Turek discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

The dispatch mechanism preferably maintains a database (reference numeral 43 in FIG. 5) of information derived from prior events that is useful in determining how a particular event may be diagnosed and corrected. ***Dispatch mechanism also preferably includes the event "correlator" (or rule base) (reference numeral 45 in FIG. 5) that uses the information in the database to help determine the nature and potential location of the event.*** As an example, a particular "event" may comprise a plurality of alarms generated by a number of resources in a particular area of the network. By parsing such information through the event correlator/database, the dispatch mechanism may make a decision about the cause of the event. In this sense, the information received by the dispatch mechanism provides a clue regarding how the new event should be addressed.

Turek at 7:32-47.

1125. Further examples include Turek at 8:66-9:4 (“As has been previously noted, the particular error and events associated therewith are useful in selecting the type of agent best suited for the diagnostic procedure. This is sometimes referred to herein as ‘event driven’ agent dispatch, because ***the particular software tasks that comprise the agent are determined by some characteristic of the event.***”); Turek at 7:58-65, 8:18-32, 9:11-17, 6:66-7:14, 9:21-29, 7:34-48, Fig. 5, Fig. 6.

**3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1126. Turek discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

***Thus, when a network error or "fault" is reported whose cause and location are not apparent or readily ascertainable, an appropriate agent is identified and dispatched to determine this information.*** Preferably, the agent is dispatched to the actual node in the network at which the fault condition occurs. As will be seen, the particular error, as well as other



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

associated events, generally provide a "clue" or clues regarding the network location to which the agent should be sent, as well as the type of agent to send. If the agent does not find the fault at the initial location to be examined, the agent then migrates through the network to locate the error. ***The agent preferably chooses its path through the network based on the information received at the dispatching location, as well as information gleaned from each examined location.***

Turek at 5:43-57.

1127. Further examples include Turek at 2:47-51 ("The software agent is then deployed into the computer network, for example, to determine a cause and location of the given event. When the software agent is received at a given node, the method determines whether the event originated from the node."); Turek at 8:43-50, 3:51-64, 4:41-49, 5:51-60, Fig. 6.

1128. To the extent this limitation is not disclosed by Turek, it would have been obvious to combine Turek with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

- b. *constructing a topological representation of the computer network from the information.*

1129. Turek discloses this element of claim 3, "constructing a topological representation of the computer network from the information." '730 Pat. at Cl. 3. For example:

Preferably, the managed environment (ME) is logically broken down into a series of loosely-connected managed regions (MR) 12, each with its own management server 14 for managing local resources with the MR. The network typically will include other servers (not shown) for carrying out other distributed network functions. These include name servers, security servers, file servers, threads servers, time servers and the like. Multiple servers 14 coordinate activities across the enterprise and permit remote site management and operation. ***Each server 14 serves a number of gateway machines 16, each of which in turn support a plurality of endpoints 18. The server 14 coordinates all activity within the MR using a terminal node manager 20.***

Turek at 3:51-64.

1130. Further examples include Turek at 4:41-49 ("The server is the top-level authority over all gateway and endpoints. The server maintains an endpoint list, which keeps track of every endpoint in a managed region. This list preferably contains all information necessary to uniquely identify and manage endpoints including, without limitation, such information as name, location, and machine type. ***The server also maintains the mapping between endpoint and gateway, and this mapping is preferably dynamic.***"); Turek at 5:51-60, 8:44-50.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1131. To the extent this limitation is not disclosed by Turek, it would have been obvious to combine Turek with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1132. Turek discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

If the agent does not find the fault at the initial location to be examined, the agent then migrates through the network to locate the error. ***The agent preferably chooses its path through the network based on the information received at the dispatching location, as well as information gleaned from each examined location.*** As will be seen, the particular "path" typically varies as the software agent migrates through the network because information gleaned from a particular node may redirect the agent in some given manner.

Turek at 5:51-60.

1133. Further examples include Turek at 3:51-64, 4:41-49.

1134. To the extent this limitation is not disclosed by Turek, it would have been obvious to combine Turek with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

1135. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek. (Afek at pp. 200-204).

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1136. Turek discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

As will be seen, the particular error, as well as other associated events, generally provide a "clue" or clues regarding the network location to which the agent should be sent, as well as the type of agent to send. If the agent does not find the fault at the initial location to be examined, the agent then migrates through the network to locate the error. The agent preferably chooses its path through the network based on the information received at the dispatching location, as well as information gleaned from each examined location. As will be seen, the particular "path" typically varies as the software agent migrates through the network because information

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

gleaned from a particular node may redirect the agent in some given manner.

Turek at 5:48-60.

1137. Further examples include Turek at 1:65-67 (“It is a primary object of this invention to *automatically diagnose faults or other events that occur in a large, distributed computer network.*”); Turek at 3:47-51, 8:20-33, 3:51-64, 4:41-49, 5:51-60, Fig. 1.

1138. To the extent this limitation is not disclosed by Turek, it would have been obvious to combine Turek with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

1139. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek. (Afek at pp. 200-204).

6. ***Claim 7***

a. *A computer network, comprising:*

1140. Turek discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

It is a primary object of this invention to automatically *diagnose faults or other events that occur in a large, distributed computer network.*

Turek at 1:65-67.

1141. Further examples include Turek at 3:37-40 (“**FIG. 5 is a distributed computer network environment** having a management infrastructure for use in carrying out the preferred method of the present invention[.]”); Turek at 3:51-64, 4:42-49, 5:51-60, 7:34-41, 7:58-65, Fig. 5.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1142. Turek discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

1143. Turek discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- d. *is able to communicate with other software agents in the computer network*

1144. Turek discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

1145. Turek discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

- f. *is able to clone itself;*

1146. Turek discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

- g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1147. Turek discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

For illustrative purposes, the method is implemented in the large, distributed enterprise environment shown in FIG. 5, although this is not a limitation of the invention. In this example, the manager 14 includes the dispatch mechanism 35 having a set of software agents 37 associated therewith. Alternatively, dispatch mechanism 35 may include a set of configurable software tasks 39 from which one or more agents are constructed. Manager 14 preferably also includes a database 43 and an event correlator 45, for the purposes described below. ***The dispatch mechanism itself may be distributed across multiple nodes. Each of the gateway nodes 16 and each of the terminal nodes 18 (or some defined subset thereof) include a runtime engine 41 that has been downloaded to the particular node via a distribution service. The engine 41 provides a runtime environment for the software agent.***

Turek at 5:62-6:10.

1148. Further examples include Turek at 8:53-58 (“Thus, in the preferred embodiment, the particular software agent is deployed into the network environment by the dispatch mechanism. The dispatch mechanism is usually supported by the system manager 14, although it may also be located at a particular gateway node 16 or it may exist as a standalone machine.”); Turek at 2:63-66, 5:3-18, 10:28-42, Fig. 5.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1149. NetFuel contends an operating system “stored in permanent or semi-permanent memory” and “providing resources such as access to processing and memory resources” satisfies this limitation. *See, e.g.,* Ex. B to NetFuel’s 8/2/18 infringement contentions, p. 98. Under this interpretation, Turek also discloses this limitation. *See* Turek at 2:63-66, 5:3-18, 10:28-42.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1150. Turek discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

1151. Turek discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

1152. Turek discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1153. Turek discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

1154. Turek discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1155. Turek discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1156. Turek discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

Referring now to FIG. 2, each gateway machine 16 runs a server component 22 of a system management framework. The server component 22 is a multi-threaded runtime process that comprises several components: an object request broker or "ORB" 21, an authorization service 23, object location service 25 and basic object adaptor or "BOA" 27. Server component 22 also includes an object library 29. ***Preferably, the ORB 21 runs continuously, separate from the operating system, and it communicates with both server and client processes through separate stubs and skeletons via an interprocess communication (IPC) facility 19. In particular, a secure remote procedure call (RPC) is used to invoke operations on remote objects.*** Gateway machines 16 also includes an operating System 15 and a threads mechanism 17.

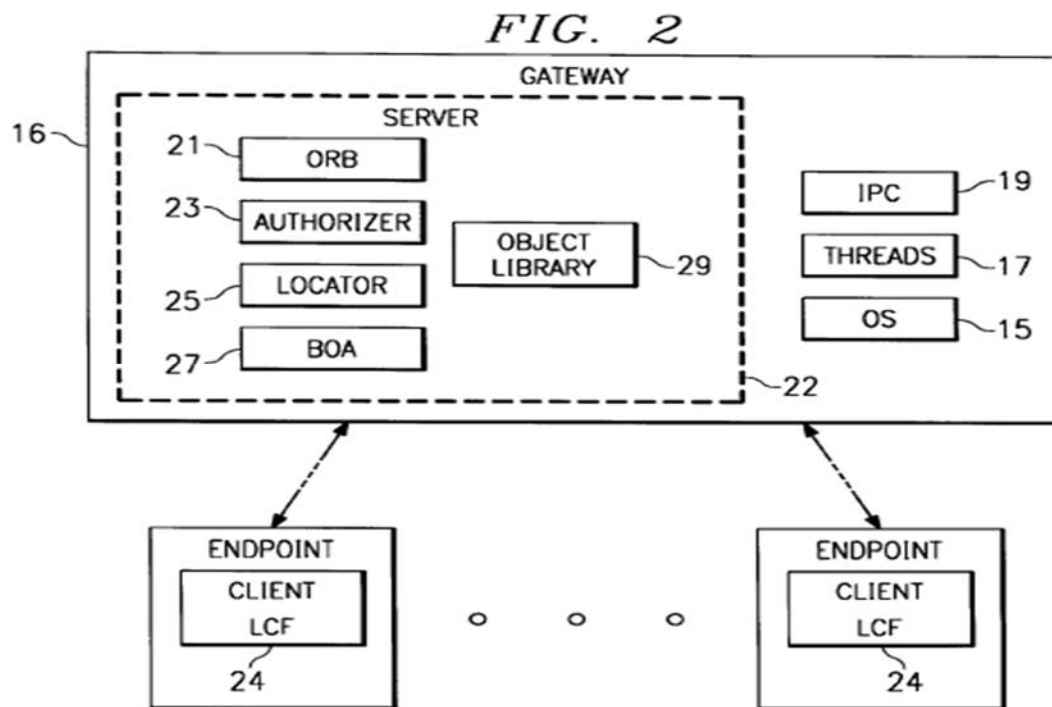
Turek at 3:65-4:12.

1157. Further examples include Turek at 2:66-3:3 (“As noted above, when the event is a fault, the software agent locates the fault and attempts to rectify it. ***If necessary, the software agent***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*may obtain additional code from the dispatch mechanism or some other network source. Such additional code may be another software agent”*); Turek at 6:38-52, 8:1-13, 9:21-29, Fig. 2.

1158. To the extent this limitation is not disclosed by Turek, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 11.0.



9. **Claim 12**

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1159. Turek discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

Referring now to FIG. 2, each gateway machine 16 runs a server component 22 of a system management framework. The server component 22 is a multi-threaded runtime process that comprises several components: an object request broker or “ORB” 21, an authorization service 23, object location service 25 and basic object adaptor or “BOA” 27. Server component 22 also includes an object library 29. Preferably, the ORB 21 runs continuously, separate from the operating system, and it communicates with both server and client processes through separate stubs and skeletons via an interprocess communication (IPC) facility 19. ***In particular, a secure remote procedure call (RPC) is used to invoke operations on remote***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*objects.* Gateway machines 16 also includes an operating System 15 and a threads mechanism 17.

Turek at 3:65-4:12.

1160. Further examples include Turek at Fig. 2.

1161. To the extent this limitation is not disclosed by Turek, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 12.0.

**B. Anticipation by and/or Obviousness in View of Great Britain Patent No. 2 363 284, issued to Goldman, et al. on 12/12/2001.**

1162. As explained in detail below and in the chart attached as Ex. A-3, Goldman anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

1163. Goldman discloses the preamble of claim 1, "[a] method of managing a computer network, comprising." '730 Pat. at Cl. 1. For example:

The present invention relates generally to ***networks***, more particularly to ***network management***, and even more particularly to policy-based network management.

Goldman at p. 1, lns. 7-8.

1164. Further examples include Goldman at p.6, lns. 4-9, p. 13, lns. 14-20, p. 13, lns. 21-28; Goldman at Fig. 4A, Fig. 4B; Goldman at Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1165. Goldman discloses this element of claim 1, "assigning a goal to a software [agent], wherein the software agent has its own runtime environment." '730 Pat. at Cl. 1. For example:

In two stage commitment, a first stage comprises ***the programming of the policy into the target or onto a policy configuration agent***, while a second stage comprises the activation of the policy on the target. Prior to activation the policy resides on the target or on the policy configuration agent but is not active in the operation of the target.

Goldman at p. 5, lns. 2-6.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1166. Further examples include Goldman at p. 13, lns. 21-28, p. 6, lns. 4-9, p. 5, lns. 9-14, p. 11, lns. 22-28, p. 14, lns. 11-19, p. 4, lns. 3-9, p. 15, lns. 11-19; Goldman at Fig. 3 (showing policy deployment to the target), Fig. 4B (showing a system for policy management by the server program for the target), Fig. 5B (showing a flow chart of policy deployment to the target with one stage policy commitment), Fig. 6B (showing a flow chart of policy deployment to the target).

c. *is able to communicate with other software agents in the computer network;*

1167. Goldman discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

Figure 5B is another flow chart of policy deployment **300** to the target **110** with one stage of policy commitment **320** as described in various representative embodiments of the present patent document. In a manner similar to that of figure 3, in block 515 the server program 410 assigns policy 120 to the target 110. Block 515 then transfers control to block **525**. In block **525 the server program 410 transfers policy 120 to the policy configuration agent 450. The policy configuration agent 450 translates the policy 120 as received from the server program 410** into policy 120 configuration specific to the target 110. Block 525 then transfers control to block 535.

Goldman at p. 14, lns. 11-19.

1168. To the extent this limitation is not expressly disclosed by Goldman, it would have been obvious to a person of ordinary skill in the art, because it was well known at the time that an agent could communicate with other agents.

d. *is capable of perceiving its own state*

1169. Goldman discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

Figure 4B is a drawing of another system 402 for policy 120 management by the server program 410 for the target 110 as described in various representative embodiments of the present patent document. In figure 4B, **the server program 410 transfers policy 120 to a policy configuration agent 450 which in turn installs the policy 120 onto the target 110. The policy configuration agent 450 translates the policy 120** as received from the server program 410 into policy 120 configuration specific to the target 110. The policy configuration agent 450 is typically a software program operating on a computer on the network 220.

Goldman at p. 13, lns. 21-28.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1170. Further examples include Goldman at p. 14, lns. 11-19, p. 14, lns. 25-28; Goldman at Fig. 4B (showing a system for policy management by the server program for the target), Fig. 5B (showing a flow chart of policy deployment to the target with one stage policy commitment).

e. *and is able to clone itself,*

1171. Goldman discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

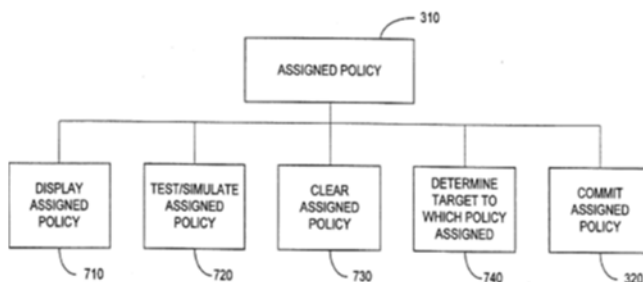


FIG.7

Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1172. Goldman discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

In two stage commitment, a first stage comprises ***the programming of the policy into the target or onto a policy configuration agent***, while a second stage comprises the activation of the policy on the target. Prior to activation the policy resides on the target or on the policy configuration agent but is not active in the operation of the target.

Goldman at p. 5, lns. 2-6.

Figure 5B is another flow chart of policy deployment 300 to the target 110 with one stage of policy commitment 320 as described in various representative embodiments of the present patent document. In a manner similar to that of figure 3, in block 515 the server program 410 assigns policy 120 to the target 110. Block 515 then transfers control to block 525. In block ***525 the server program 410 transfers policy 120 to the policy configuration agent 450. The policy configuration agent 450 translates the policy 120 as received from the server program 410 into policy 120 configuration specific to the target 110.*** Block 525 then transfers control to block 535.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Goldman at p. 14, lns. 11-19.

1173. Further examples include Goldman at p. 14, lns. 4-8, p. 15, lns. 11-19; Goldman at Fig. 5B (showing a flow chart of policy deployment to the target with one stage policy commitment), Fig. 6B (showing a flow chart of policy deployment to the target).

g. *monitoring the computer network;*

1174. Goldman discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

The present invention relates generally to **networks**, more particularly to **network management**, and even more particularly to policy-based network management.

Goldman at p. 1, lns. 7-8.

1175. Further examples include Goldman at p. 4, lns. 3-9, p. 4, lns. 18-20.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1176. Goldman discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

The present patent document relates to a novel method for deployment of policy to a target connected to a network for the purpose of controlling the actions of that target based upon certain predefined conditions. In representative embodiments, methods are disclosed for creating another step in the policy deployment process within which **policies can be created, tested, changed, and deleted** prior to their transfer to the policy configuration agents of the targets to which it is intended that they will eventually be deployed.

Goldman at p. 4, lns. 3-9.

1177. Further examples include Goldman at p. 2, ln 29 - p.3, ln. 3, p. 16, lns. 5- 25, p. 5, ln. 25- p. 6, ln. 3, p.6, lns. 4-9; Goldman at Fig. 3 (showing policy deployment to the target), Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***prediction algorithm*

1178. Goldman discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

The second benefit mentioned above is that there are target-specific operations that users might want to perform on a particular policy/target pair. One clear example is to validate the policy for a particular target. This validation step is important because a target may support a given policy type and yet not support all possible condition types for that policy type or the given policy may conflict with other existing target configuration information. ***If users can validate the policy for the intended target before committing the policy, they can avoid problems like leaving the target incorrectly configured or un-configured with respect to OoS.*** Another example of a target-specific operation would be policy simulation.

Goldman at p. 5, ln. 25- p. 6, ln. 3.

1179. Further examples include Goldman at p.6, lns. 4-9, p. 16, lns. 5- 25; Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1180. Goldman discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

The second benefit mentioned above is that there are target-specific operations that users might want to perform on a particular policy/target pair. One clear example is to validate the policy for a particular target. This validation step is important because a target may support a given policy type and yet not support all possible condition types for that policy type or the given policy may conflict with other existing target configuration information. ***If users can validate the policy for the intended target before committing the policy, they can avoid problems like leaving the target incorrectly configured or un-configured with respect to OoS.*** Another example of a target-specific operation would be policy simulation.

Goldman at p. 5, ln. 25- p. 6, ln. 3.

1181. Further examples include Goldman at p. 16, lns. 5- 25; Goldman at Fig. 7.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1182. Goldman discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Operation 320 of figure 7, as in figure 3, commits assigned policy 120 for the given target 110. The system 400 allows the user to commit the assigned policy 120 on the given target 110. ***This operation moves the assigned policy 120 into the committed state on the target 110, overwriting the target's 110 previously committed policy 120 and clearing the target's 110 assigned policy 120.*** This operation affects both the stored relationships for the target 110, i.e., assigned and committed policy 120, as well as the target's 110 configuration, i.e., changing the installed policy 120 on the target 110.

Goldman at p. 16, ln. 26 - p. 17

Operation 720 of figure 7 ***tests/simulates assigned policy 120*** for a given target 110. Operation 730 of figure 7 clears assigned policy 120 for a given target 110. The system 400 allows the user to clear the assigned policy 120 for a given target 110. In this case, the system clears the assigned policy 120 relationship for that target 110. ***Operation 740 of figure 7 determines to which targets 110 the policy 120 is assigned.*** The system 400 allows the user to see to which targets 110 a given policy 120 is assigned. This operation is supported by a query which searches through the 25 assignment relationships for entries which include a reference to the given policy 120.

Goldman at p. 16, lns 17-25.

1183. Further examples include Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1184. Goldman discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

Deploying policy involves moving the policy onto the target or target configuration agent, translating the policy into target-specific configuration, and loading this configuration.

Goldman at p. 2, lns. 7-9.

1185. Further examples include Goldman at p. 13, lns. 21-28; Goldman at Fig. 4B (showing a system for policy management by the server program for the target).

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*expressed as a policy.*

1186. Goldman discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

In two stage commitment, a first stage comprises ***the programming of the policy into the target or onto a policy configuration agent***, while a second stage comprises the activation of the policy on the target. Prior to activation the policy resides on the target or on the policy configuration agent but is not active in the operation of the target.

Goldman at p. 5, lns. 2-6.

1187. Further examples include Goldman at p. 6, lns. 4-9, p. 11, lns. 22-28, p. 2, lns. 7-9, p. 13, lns. 21-28; Goldman at Fig. 3 (showing policy deployment to the target), Fig. 4B (showing a system for policy management by the server program for the target).

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1188. Goldman discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

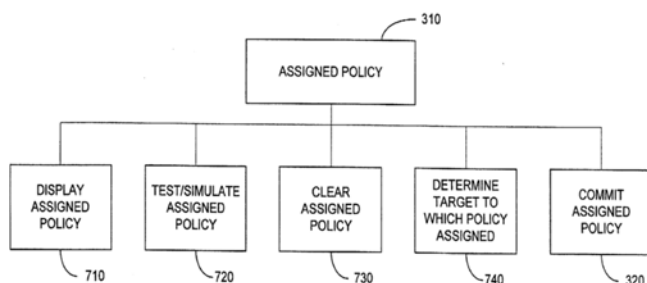
One clear example is to ***validate the policy 120 for a particular target 110***. This validation step is important because the target 110 may support a given policy 120 type and yet not support all possible condition types for that policy 120 type or the given policy 120 may conflict with other existing target 110 configuration information.

Goldman at p. 12, lns. 19-23.

1189. Further examples include Goldman at p. 6, lns. 10-14, p. 10, lns. 4-6, p. 10, lns. 19-26.

- b. *constructing a topological representation of the computer network from the information.*

1190. Goldman discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****FIG.7**

Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1191. Goldman discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

Figure 4B is a drawing of another system 402 for policy 120 management by the server program 410 for the target 110 as described in various representative embodiments of the present patent document. In figure 4B, the server program 410 transfers policy 120 to a policy configuration agent 450 which in turn installs the policy 120 onto the target 110. The policy configuration agent 450 translates the policy 120 as received from the server program 410 into policy 120 configuration specific to the target 110. The policy configuration agent 450 is typically a software program operating on a computer on the network 220.

Goldman at p. 13, lns. 21-28.

1192. Further examples include Goldman at Fig. 4B (showing a system for policy management by the server program for the target).

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1193. Goldman discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 4B is a drawing of another system 402 for policy 120 management by the server program 410 for the target 110 as described in various representative embodiments of the present patent document. In figure 4B, the server program 410 transfers policy 120 to a policy configuration agent 450 which in turn installs the policy 120 onto the target 110. The policy configuration agent 450 translates the policy 120 as received from the server program 410 into policy 120 configuration specific to the target 110. The policy configuration agent 450 is typically a software program operating on a computer on the network 220.

Goldman at p. 13, lns. 21-28.

1194. Further examples include Goldman at p. 16, ln. 26 - p. 17 ln. 2; p. 16, lns. 17-21; Goldman at Fig. 4B (showing a system for policy management by the server program for the target), Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

6. ***Claim 7***

a. *A computer network, comprising:*

1195. Goldman discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

The present invention relates generally to ***networks***, more particularly to network management, and even more particularly to policy-based network management.

Goldman at p. 1, lns. 7-8.

1196. Further examples include Goldman at p. 13, lns. 14-20, p. 13, lns. 21-28, p. 4, lns. 15-16; Goldman at Fig. 4A (showing a system for policy management by a server program for the target); Fig. 4B (showing a system for policy management by the server program for the target); Goldman at Claim 1.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1197. Goldman discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

1198. Goldman discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

1199. Goldman discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

1200. Goldman discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

1201. Goldman discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1202. Goldman discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

Figure 4B is a drawing of another system 402 for policy 120 management by the server program 410 for the target 110 as described in various representative embodiments of the present patent document. In figure 4B, the server program 410 transfers policy 120 to a policy configuration agent 450 which in turn installs the policy 120 onto the target 110. The policy configuration agent 450 translates the policy 120 as received from the server program 410 into policy 120 configuration specific to the target 110. ***The policy configuration agent 450 is typically a software program operating on a computer on the network 220.***

Goldman at p. 13, lns. 21-28.

1203. Further examples include Goldman at Claim 2; Goldman at Fig. 4B (showing a system for policy management by the server program for the target).

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*thereby to determine an optimal policy for the computer network*

1204. Goldman discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

1205. Goldman discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

1206. Goldman discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1207. Goldman discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1208. Goldman discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*an operational characteristic of the network.*

1209. Goldman discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1210. Goldman discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

Figure 5B is another flow chart of policy deployment 300 to the target 110 with one stage of policy commitment 320 as described in various representative embodiments of the present patent document. In a manner similar to that of figure 3, in block 515 the server program 410 assigns policy 120 to the target 110. Block 515 then transfers control to block 525. In block **525 the server program 410 transfers policy 120 to the policy configuration agent 450. The policy configuration agent 450 translates the policy 120 as received from the server program 410** into policy 120 configuration specific to the target 110. Block 525 then transfers control to block 535.

Goldman at p. 14, lns. 11-19.

1211. Further examples include Goldman at Fig. 5B (showing a flow chart of policy deployment to the target with one stage policy commitment).

9. ***Claim 12***

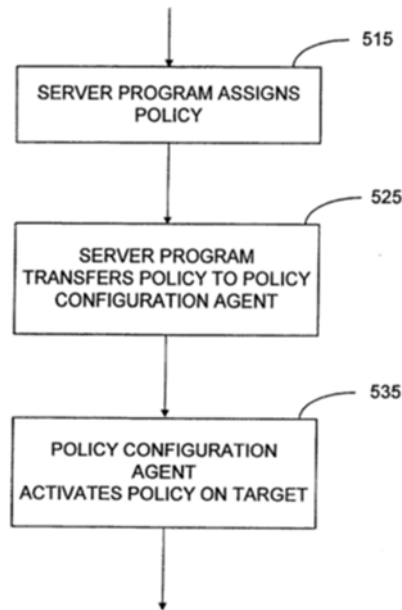
- a. *The computer network of claim 11, wherein the communications*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism comprises a secure communications protocol.*

1212. Goldman discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:



**FIG.5B**

Goldman at Fig. 5B (showing a flow chart of policy deployment to the target with one stage policy commitment).

1213. Further examples include Goldman at Fig. 6B (showing a flow chart of policy deployment to the target).

**10. Claim 16**

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

1214. Goldman discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Deploying policy involves moving the policy onto the target or target configuration agent, translating the policy into *target-specific configuration*, and loading this configuration.

Goldman at p. 2, lns. 7-10.

1215. Further examples include Goldman at p. 2, ln 29 - p.3, ln. 3, p. 5, lns. 9-14, p. 5, lns. 15- 18, p. 5, lns. 25-30, p. 14, lns. 11-19; Goldman at Fig. 5B (showing a flow chart of policy deployment to the target with one stage policy commitment).

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1216. Goldman discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

The present patent document relates to a novel method for deployment of policy to a target connected to a network for the purpose of controlling *the actions of that target based upon certain predefined conditions*. In representative embodiments, methods are disclosed for creating another step in the policy deployment process within which policies can be created, tested, changed, and deleted prior to their transfer to the policy configuration agents of the targets to which it is intended that they will eventually be deployed.

Goldman at p. 4, lns. 3-9.

1217. Further examples include Goldman at p. 5, lns. 2-6.

12. ***Claim 18***

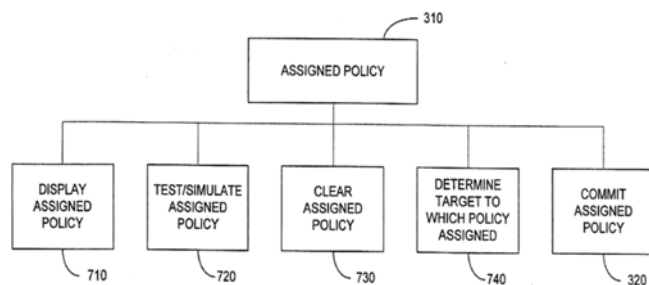
- a. *The computer network of claim 17, wherein the operation is*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*selected from the group comprising of spawn, kill and suspend.*

1218. Goldman discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:



**FIG.7**

Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

**13. Claim 19**

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

1219. Goldman discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

Operation 310 of figure 7, as in figure 3, assigns policy 120. ***The system 400*** allows the user to assign the policy 120 to the target 110 on a per target 110 basis, i.e., given the target 110, ***present the list of possible policies 120 so that one can be assigned, or on a per policy 120 basis, i.e., given the policy 120, present the list of targets 110 which support the policy's 120 type so the policy 120 can be assigned to one of them.*** This operation will create and store the assignment relationship based on the target 110 as described above.

Goldman at p. 16, lns. 5-11.

**14. Claim 21**

- a. *The computer network of claim 7, further comprising a policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*database to store the policy for the agent.*

1220. Goldman discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

Figure 4A is a drawing of a system 400 for policy 120 management by a server program 410 for the target 110 as described in various representative embodiments of the present patent document. A console 430 connected to the server program 410 provides the user interface to enable the assignment of policy 120 to the appropriate targets 110 prior to commitment. ***The policy 120 is typically stored in a memory 445 located on a computer program storage medium 447*** connected to the server program 410, all of which could be located on a computer 405.

Goldman at p. 13, lns. 14-20.

1221. Further examples include Goldman at Fig. 4A (showing a system for policy management by a server program for the target), Fig. 4B (showing a system for policy management by the server program for the target).

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1222. Goldman discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

Figure 4A is a drawing of a system 400 for policy 120 management by a server program 410 for the target 110 as described in various representative embodiments of the present patent document. A console 430 connected to the server program 410 ***provides the user interface to*** enable the assignment of policy 120 to the appropriate targets 110 prior to commitment. The policy 120 is typically stored in a memory 445 located on a computer program storage medium 447 connected to the server program 410, all of which could be located on a computer 405.

Goldman at p. 13, lns. 14-20.

1223. Further examples include Goldman at Fig. 4A (showing a system for policy management by a server program for the target).

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*a Dijkstra Self Stabilization Algorithm.*

1224. Goldman discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1225. Goldman discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The present patent document relates to a novel method for deployment of policy to a target connected to a network for the purpose of controlling the actions of that target based upon certain predefined conditions. In representative embodiments, methods are disclosed for creating another step in the policy deployment process within which ***policies can be created, tested, changed, and deleted*** prior to their transfer to the policy configuration agents of the targets to which it is intended that they will eventually be deployed.

Goldman at p. 4, lns. 3-9.

1226. Further examples include Goldman at p. 2, ln 29 - p.3, ln. 3, p. 16, lns. 5- 25; Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy).

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

1227. Goldman discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

Note that in one stage policy transfer of the policy 120 from the server program 410 to the policy configuration agent 450 and subsequent loading and activating policy 120 by the policy configuration agent 450 on the target 110 occurs as substantially ***without further user input***.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Goldman at p. 14, lns. 25-28.

1228. Further examples include Goldman at p. 16, lns. 12- 25, p. 14, lns. 11-19; Goldman at Fig. 7 (showing a block diagram of operations that can be performed on the assigned policy), Fig. 5B (showing a flow chart of policy deployment to the target with one stage policy commitment).

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

1229. Goldman discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

***A computer program storage medium [447] readable by a computer,*** tangibly embodying a computer program of instructions executable by the computer to perform method steps, the method steps comprising: assigning a policy [120] to a target [110], providing the policy [120] specifies conditional action implementable on the target [110], providing the target is a resource on a network [220], and providing policy [120] assignment comprises association of the policy [120] with the target [110] prior to policy [120] reconfiguration of the target [110]; and activating the policy [120] on the target [110], providing the policy [120] has been activated when target [110] actions comply with the policy [120].

Goldman at Claim 6.

1230. Further examples include Goldman at p. 13, lns. 14-20, Fig. 4A (showing a system for policy management by a server program for the target), p. 13, lns. 21-29.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1231. Goldman discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

1232. Goldman discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

d. *is capable of perceiving its own state; and*

1233. Goldman discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself; and*

1234. Goldman discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task and*

1235. Goldman discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *wherein the goal is a programmatic expression of a predefined task for the software agent*

1236. Goldman discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1237. Goldman discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

1238. Goldman discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1239. Goldman discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

1240. Goldman discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1241. Goldman discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1242. Goldman discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

1243. Goldman discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

1244. Goldman discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1245. Goldman discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1246. Goldman discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1247. Goldman discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1248. I expect to testify that Goldman anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Goldman discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-3.

**C. Anticipation by and/or Obviousness in View of United States Patent No. 6,671,724, filed 3/21/2000 and issued to Pandya, et al. on 12/30/2003.**

1249. As explained in detail below and in the chart attached as Ex. A-4, Pandya anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

1250. Pandya discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

Software, systems and ***methods for managing a distributed network environment including a plurality of computers interconnected by a network link***, where at least some of the computers include a layered communications protocol stack for providing a data interface between an application program and the network link, the communications stack having a transport protocol layer for providing an end-to-end communications connection. The invention includes a control module and a plurality of agent modules, each agent being associated with one of the computers and adapted to dynamically monitor the associated computer at a data transmission point between an application program running on the computer and the transport protocol layer and repeatedly communicate with the control module in order to effect management of the distributed network system. The invented software, systems and methods may also include a messaging feature for providing users, IT personnel, or various management systems with informative messages concerning network conditions and network resources.

Pandya at Abstract.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1251. Pandya discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

Software, systems and methods for managing a distributed network environment including a plurality of computers interconnected by a network link, where at least some of the computers include a layered communications protocol stack for providing a data interface between an application program and the network link, the communications stack having a transport protocol layer for providing an end-to-end communications connection. The invention includes a control module and a ***plurality of agent modules, each agent being associated with one of the computers*** and adapted to dynamically monitor the associated computer at a data transmission point between an application program running on the computer and the transport protocol layer and repeatedly communicate with the control module in order to effect management of the distributed network system. The invented software, systems and methods may also include a messaging feature for providing users, IT personnel, or various management systems with informative messages concerning network conditions and network resources.

Pandya at Abstract.

1252. Further examples include Pandya at 4:31-46, 7:27-39, 9:65-10:11, 14:35-44; Pandya at Figs. 4, 6.

- c. *is able to communicate with other software agents in the computer network;*

1253. Pandya discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

An embodiment of the agent module is depicted in FIG. 9. As shown, agent 70 may include a redirector module 130, a traffic control module 132, an administrator module 134, a DNS module 136, a popapp module 138, a ***message broker module 140***, a system services module 142, and a popapp 144. Redirector module 130 intercepts winsock API calls made by applications running on networked devices such as the client computers depicted in FIGS. 2 and 3. Redirector module 130 then hands these calls to one or more of the other agent components for processing. As discussed with reference to FIGS. 6-8, redirector module is positioned to allow the agent to monitor data at a data transmission point between an application program running on the device and the transport layer of the communications stack. Depending on the configuration of the agent and control point, the intercepted winsock calls may be rejected, changed, or passed on by agent 70.

Pandya at 10:66-11:15.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1254. Further examples include Pandya at 13:65-14:5; Pandya at Figs. 9, 5.

d. *is capable of perceiving its own state*

1255. Pandya discloses this element of claim 1, “is capable of perceiving its own state.”

’730 Pat. at Cl. 1. For example:

Administrator module 134 interacts with various other agent modules, maintains and provides network statistics, and provides an interface for centrally configuring agents and other components of the invented system. ***With regard to agent configuration, administrator module 134 interfaces with configuration utility 106 (shown in FIGS. 5 and 13-16), in order to configure various agent parameters.*** Administrator module 134 also serves as a repository for local reporting and statistics information to be communicated to the control points. Based on information obtained by other agent modules, administrator module 134 maintains local information regarding accessed servers, DNS servers, gateways, routers, switches, applications and other resources. This information is communicated on request to the control point, and may be used for network planning or to dynamically alter the behavior of agents. ***In addition, administrator module 134 stores system policies and/or components of policies, and provides policy data to various agent components as needed to implement and enforce the policies.*** Administrator module 134 also includes support for interfacing the invented system with standardized network management protocols and platforms.

Pandya at 13:20-41.

e. *and is able to clone itself,*

1256. Pandya discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at

Cl. 1. For example:

Once the control point obtains the status information, the control point reports the information to all of the agents in its domain, and instructs the agents how to handle further client requests involving the resource, as indicated at steps S108 and S110. In the event that the target resource is down, underperforming or otherwise unavailable, the instructions given to the agents will depend on whether an alternate resource is available. The control point stores dynamically updated lists of alternate available resources. If an alternate resource is available, the instructions provided to the agent may include an instruction to transparently redirect the request to an alternate resource, as shown in step S108. For example, if the control point knows of a server that mirrors the data of another server that has gone down, client requests to the down server can simply be redirected to the mirror server. Alternatively, if no alternate resource is available, the agent can be instructed to provide a user message in the event of an access attempt, as seen in step S110.

Pandya at 19:7-24.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1257. Further examples include Pandya at 19:40-44; Pandya at Figs. 12.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1258. Pandya discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

For example, successful businesses often strive to provide excellent customer services. *This underlying business goal can be translated into many different policies defining how network resources are to be used. One example of such a policy would be to prevent or limit access to non-business critical applications when performance of business critical applications is degraded beyond a threshold point. Another example would be to use QoS techniques to provide a guaranteed or high level of service to e-commerce applications. Yet another example would be to dynamically increase the network bandwidth allocated to a networked computer whenever it is accessed by a customer. Also, bandwidth for various applications might be restricted during times when there is heavy use of network resources by customers.*

*Control points 72 would access these policies and provide policy data to agents 70. Agents 70 and control points 72 would communicate with each other and monitor the network to determine how many customers were accessing the network, what computers the customer(s) were accessing, and what applications were being accessed by the customers. Once the triggering conditions were detected, the agents and control points would interact to re-allocate bandwidth, provide specified service levels, block or restrict various non-customer activities, etc.*

*Another example of policy-based management would be to define an optimum specification of network resources or service levels for particular types of network tasks.* The particular policies would direct the management entities to determine whether the particular task was permitted, and if permitted, the management entities would interact to ensure that the desired level of resources was provided to accomplish the task. If the optimum resources were not available, the applicable policies could further specify that the requested task be blocked, and that the requesting user be provided with an informative message detailing the reason why the request was denied. Alternatively, the policies could specify that the user be provided with various options, such as proceeding with the requested task, but with sub-optimal resources, or waiting to perform the task until a later time.

For example, continuous media applications such as IP telephony have certain bandwidth requirements for optimum performance, and are particularly sensitive to network jitter and delay. *Policies could be written to specify a desired level of service, including bandwidth requirements and threshold levels for jitter and delay, for client computers attempting to run IP telephony applications. The policies would further direct the agents and control modules to attempt to provide the specified level of service.* Security checking could also be included to ensure that the



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

particular user or client computer was permitted to run the application. In the event that the specified service level could not be provided, the requesting user could be provided with a message indicating that the resources for the request were not available. The user could also be offered various options, including proceeding with a sub-optimal level of service, placing a conventional telephone call, waiting to perform the task until a later time, etc.

The software, system and methods of the present invention may be used to implement a wide variety of system policies. *The policy rules and conditions may be based on any number of parameters, including IP source address, IP destination address, source port, destination port, protocol, application identity, user identity, device identity, URL, available device bandwidth, application profile, server profile, gateway identity, router identity, time-of-day, network congestion, network load, network population, available domain bandwidth and resource status*, to name but a partial list. *The actions taken when the policy conditions are satisfied can include blocking network access, adjusting service levels and/or bandwidth allocations for networked devices, blocking requests to particular URLs, diverting network requests away from overloaded or underperforming resources, redirecting network requests to alternate resources and gathering network statistics.*

*Some of the parameters listed above may be thought of as "client parameters," because they are normally evaluated by an agent monitoring a single networked client device. These include IP source address, IP destination address, source port, destination port, protocol, application identity, user identity, available device bandwidth and URL. Other parameters, such as application profile, server profile, gateway identity, router identity, time-of-day, network congestion, network load, network population, available domain bandwidth and resource status may be thought of as "system parameters" because they pertain to shared resources, aggregate network conditions or require evaluation of data from multiple agent modules.* Despite this, there is not a precise distinction between client parameters and system parameters. Certain parameters, such as time-of-day, may be considered either a client parameter or a system parameter, or both.

Pandya at 8:7-9:31.

g. *monitoring the computer network;*

1259. Pandya discloses this element of claim 1, "monitoring the computer network." '730

Pat. at Cl. 1. For example:

Referring now to FIGS. 6-9, the agent module will be more particularly described. *The basic functions of the agent module are monitoring the status and activities of its associated client*, server, pervasive computing device or other computing device, communicating this information to one or more control points, *enforcing system policies under the direction of the control points*, and providing messages to network users and administrators concerning network conditions. FIGS. 6-8 are conceptual depictions of networked computing devices, and show how the agent software is

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

associated with the networked devices relative to layered protocol software used by the devices for network communication.

Pandya at 9:65-10:11.

1260. Further examples include Pandya at 21:17-38.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1261. Pandya discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

FIGS. 11A, 11B, 11C and 11D depict examples of various methods that may be implemented by traffic module 160 to dynamically allocate bandwidth. FIG. 11A depicts a process by which traffic module 160 determines whether any adjustments to bandwidth allocations AB are necessary...

Second, if there are any agents for which  $AB < CB$  and  $UB \cong AB$ , the allocation for those agents is modified, as seen in steps S6 and S10. The allocations for any such agent are typically increased.... Third, if there are any agents reporting bandwidth usage UB that is less than their allocation AB, as determined at step S8, then the allocation AB for such an agent is reduced for the upcoming period to free up the unused bandwidth. Steps S4, S6 and S8 may be performed in any suitable order. Collectively, these three steps ensure that certain bandwidth allocations are modified, i.e. increased or reduced, if one or more of the following three conditions are true: (1)  $AB_{total} > CB_{total}$ , (2)  $AB < CB$  and  $UB \cong AB$  for any agent, or (3)  $UB < AB$  for any agent. If none of these are true, the allocations AB from the prior period are not adjusted. Traffic module 160 modifies allocations AB as necessary at step S10. After all necessary modifications are made, the control point communicates the new allocations to the agents for enforcement during the upcoming cycle.

Pandya at 16:28-17:35.

1262. Further examples include Pandya at 17:36-59, 17:36-18:44, 18:48-19:44; Pandya at FIGS. 11A-D.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1263. Pandya discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

In addition to assisting these resource monitoring functions, server profile module 162 maintains a dynamically updated list of the servers accessed by

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agents within its domain. The server statistics may be retrieved using the configuration utility, or with a variety of other existing management platforms. ***The server statistics may be used for network planning***, or may be implemented into various system policies for dynamic enforcement by the agents and control points. For example, ***the control points and agents can be configured to divert traffic from heavily used servers or other resources***.

Pandya at 19:57-67.

1264. Further examples include Pandya at 20:1-12, 21:17-38.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1265. Pandya discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

In addition, optimum and minimum performance levels may be established for applications or other tasks using network resources. Referring again to the IP telephony example discussed above, the configuration utility may be used to ***specify a minimum threshold performance level for a networked device running the IP telephony application***. This performance level may be specified in terms of QoS performance parameters such as bandwidth, throughput, jitter, delay and loss. ***The agent module associated with the networked device would then monitor the network traffic associated with the IP telephony application to ensure that performance was above the minimum threshold. If the minimum level was not met, the control points and agents could interact to reallocate resources and provide the specified minimum service level***. Similarly, an optimum service level may be specified for various network applications and tasks. More generally, configuration utility 106 may be configured to manage system policies by providing functionality for authoring, maintaining and storing system policies, and for managing retrieval of system policies from other locations on a distributed network, such as a dedicated policy server.

Pandya at 21:17-38.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1266. Pandya discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl.

1. For example:

The agents and control points interact to control and monitor network events, track operational and congestion status of network resources, select optimum targets for network requests, ***dynamically manage bandwidth usage***, and share information about network conditions with customers, users and IT personnel.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Pandya at 4:40-46.

1267. Further examples include Pandya at 16:28-17:35, 17:36-18:44, 4:40-45, 8:16-20, 14:54-67, 19:32-44; Pandya at FIGS. 11A-D.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1268. Pandya discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

In addition, optimum and minimum performance levels may be established for applications or other tasks using network resources. Referring again to the IP telephony example discussed above, the configuration utility may be used to ***specify a minimum threshold performance level for a networked device running the IP telephony application.*** This performance level may be specified in terms of QoS performance parameters such as bandwidth, throughput, jitter, delay and loss. ***The agent module associated with the networked device would then monitor the network traffic associated with the IP telephony application to ensure that performance was above the minimum threshold. If the minimum level was not met, the control points and agents could interact to reallocate resources and provide the specified minimum service level.*** Similarly, an optimum service level may be specified for various network applications and tasks. More generally, configuration utility 106 may be configured to manage system policies by providing functionality for authoring, maintaining and storing system policies, and for managing retrieval of system policies from other locations on a distributed network, such as a dedicated policy server.

Pandya at 21:17-38.

1269. Further examples include Pandya at 12:44-63, 18:59-19:18.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1270. Pandya discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

For example, successful businesses often strive to provide excellent customer services. ***This underlying business goal can be translated into many different policies defining how network resources are to be used. One example of such a policy would be to prevent or limit access to non-business critical applications when performance of business critical applications is degraded beyond a threshold point. Another example would be to use QoS techniques to provide a guaranteed or high level of***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*service to e-commerce applications. Yet another example would be to dynamically increase the network bandwidth allocated to a networked computer whenever it is accessed by a customer. Also, bandwidth for various applications might be restricted during times when there is heavy use of network resources by customers.*

*Control points 72 would access these policies and provide policy data to agents 70. Agents 70 and control points 72 would communicate with each other and monitor the network to determine how many customers were accessing the network, what computers the customer(s) were accessing, and what applications were being accessed by the customers. Once the triggering conditions were detected, the agents and control points would interact to re-allocate bandwidth, provide specified service levels, block or restrict various non-customer activities, etc.*

*Another example of policy-based management would be to define an optimum specification of network resources or service levels for particular types of network tasks. The particular policies would direct the management entities to determine whether the particular task was permitted, and if permitted, the management entities would interact to ensure that the desired level of resources was provided to accomplish the task. If the optimum resources were not available, the applicable policies could further specify that the requested task be blocked, and that the requesting user be provided with an informative message detailing the reason why the request was denied. Alternatively, the policies could specify that the user be provided with various options, such as proceeding with the requested task, but with sub-optimal resources, or waiting to perform the task until a later time.*

For example, continuous media applications such as IP telephony have certain bandwidth requirements for optimum performance, and are particularly sensitive to network jitter and delay. *Policies could be written to specify a desired level of service, including bandwidth requirements and threshold levels for jitter and delay, for client computers attempting to run IP telephony applications. The policies would further direct the agents and control modules to attempt to provide the specified level of service.* Security checking could also be included to ensure that the particular user or client computer was permitted to run the application. In the event that the specified service level could not be provided, the requesting user could be provided with a message indicating that the resources for the request were not available. The user could also be offered various options, including proceeding with a sub-optimal level of service, placing a conventional telephone call, waiting to perform the task until a later time, etc.

The software, system and methods of the present invention may be used to implement a wide variety of system policies. *The policy rules and conditions may be based on any number of parameters, including IP source address, IP destination address, source port, destination port, protocol, application identity, user identity, device identity, URL, available device bandwidth, application profile, server profile, gateway identity, router identity, time-of-day, network congestion, network load, network population, available domain bandwidth and resource status, to name but a partial list. The actions taken when the policy conditions are satisfied*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*can include blocking network access, adjusting service levels and/or bandwidth allocations for networked devices, blocking requests to particular URLs, diverting network requests away from overloaded or underperforming resources, redirecting network requests to alternate resources and gathering network statistics.*

*Some of the parameters listed above may be thought of as “client parameters,” because they are normally evaluated by an agent monitoring a single networked client device. These include IP source address, IP destination address, source port, destination port, protocol, application identity, user identity, available device bandwidth and URL. Other parameters, such as application profile, server profile, gateway identity, router identity, time-of-day, network congestion, network load, network population, available domain bandwidth and resource status may be thought of as “system parameters” because they pertain to shared resources, aggregate network conditions or require evaluation of data from multiple agent modules.* Despite this, there is not a precise distinction between client parameters and system parameters. Certain parameters, such as time-of-day, may be considered either a client parameter or a system parameter, or both.

Pandya at 8:7-9:31.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1271. Pandya discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

In addition to assisting these resource monitoring functions, server profile module 162 maintains a dynamically updated list of the servers accessed by agents within its domain. The server statistics may be retrieved using the configuration utility, or with a variety of other existing management platforms. ***The server statistics may be used for network planning***, or may be implemented into various system policies for dynamic enforcement by the agents and control points. For example, ***the control points and agents can be configured to divert traffic from heavily used servers or other resources.***

Pandya at 19:57-67.

1272. Further examples include Pandya at 20:1-12.

- b. *constructing a topological representation of the computer network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

1273. Pandya discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

In addition to assisting these resource monitoring functions, server profile module 162 maintains a dynamically updated list of the servers accessed by agents within its domain. The server statistics may be retrieved using the configuration utility, or with a variety of other existing management platforms. ***The server statistics may be used for network planning***, or may be implemented into various system policies for dynamic enforcement by the agents and control points. For example, ***the control points and agents can be configured to divert traffic from heavily used servers or other resources.***

Pandya at 19:57-67.

1274. Further examples include Pandya at 20:1-12.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1275. Pandya discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

In addition to assisting these resource monitoring functions, server profile module 162 maintains a dynamically updated list of the servers accessed by agents within its domain. The server statistics may be retrieved using the configuration utility, or with a variety of other existing management platforms. ***The server statistics may be used for network planning***, or may be implemented into various system policies for dynamic enforcement by the agents and control points. For example, ***the control points and agents can be configured to divert traffic from heavily used servers or other resources.***

Pandya at 19:57-67.

1276. Further examples include Pandya at 20:1-12.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1277. Pandya discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

In addition to assisting these resource monitoring functions, server profile module 162 maintains a dynamically updated list of the servers accessed by

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agents within its domain. The server statistics may be retrieved using the configuration utility, or with a variety of other existing management platforms. ***The server statistics may be used for network planning***, or may be implemented into various system policies for dynamic enforcement by the agents and control points. For example, ***the control points and agents can be configured to divert traffic from heavily used servers or other resources***.

Pandya at 19:57-67.

1278. Further examples include Pandya at 20:1-12.

6. ***Claim 7***

a. *A computer network, comprising:*

1279. Pandya discloses the preamble of claim 7, “[a] computer network, comprising.”  
’730 Pat. at Cl. 7. Pandya Figure 2 illustrates this preamble.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1280. Pandya discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

1281. Pandya discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

1282. Pandya discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

1283. Pandya discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

f. *is able to clone itself;*

1284. Pandya discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1285. Pandya discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

Referring now to FIGS. 6-9, the agent module will be more particularly described. The basic functions of the agent module are monitoring the status and activities of its associated client, server, pervasive computing device or other computing device, communicating this information to one or more control points, enforcing system policies under the direction of the control points, and providing messages to network users and administrators concerning network conditions. ***FIGS. 6-8 are conceptual depictions of networked computing devices***, and show how the agent software is associated with the networked devices relative to layered protocol software used by the devices for network communication.

Pandya at 9:65-10:11.

1286. Further examples include Pandya at 14:35-44; Pandya at Figs. 3, 6-8, 9.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1287. Pandya discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

1288. Pandya discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

1289. Pandya discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1290. Pandya discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1291. Pandya discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1292. Pandya discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1293. Pandya discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

An embodiment of the agent module is depicted in FIG. 9. As shown, agent 70 may include a redirector module 130, a traffic control module 132, an administrator module 134, a DNS module 136, a popapp module 138, **a message broker module 140**, a system services module 142, and a popapp 144. Redirector module 130 intercepts winsock API calls made by applications running on networked devices such as the client computers depicted in FIGS. 2 and 3. Redirector module 130 then hands these calls to one or more of the other agent components for processing. As discussed with reference to FIGS. 6-8, redirector module is positioned to allow the agent to monitor data at a data transmission point between an application program running on the device and the transport layer of the communications stack. Depending on the configuration of the agent and control point, the intercepted winsock calls may be rejected, changed, or passed on by agent 70.

Pandya at 10:66-11:15.

1294. Further examples include Pandya at 13:65-14:5.

9. **Claim 12**

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1295. Pandya discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

An embodiment of the agent module is depicted in FIG. 9. As shown, agent 70 may include a redirector module 130, a traffic control module 132, an administrator module 134, a DNS module 136, a popapp module 138, **a message broker module 140**, a system services module 142, and a popapp 144. Redirector module 130 intercepts winsock API calls made by applications running on networked devices such as the client computers depicted in FIGS. 2 and 3. Redirector module 130 then hands these calls to one or more of the other agent components for processing. As discussed with reference to FIGS. 6-8, redirector module is positioned to allow the agent to monitor data at a data transmission point between an application program running on the device and the transport layer of the communications stack. Depending on the configuration of the agent and control point, the intercepted winsock calls may be rejected, changed, or passed on by agent 70.

Pandya at 10:66-11:15.

1296. Further examples include Pandya at 13:65-14:5.

10. **Claim 16**

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*for a particular network component.*

1297. Pandya discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

***System services module 142 provides various support functions to the other agent components.*** First, system services module maintains dynamic lists of user profiles, server profiles, DNS server profiles, control point connections and other data. The system services module also provides a tracing capability for debugging, and timer services for use by other agent components. ***System services module may also be configured with a library of APIs to interface the agent with the operating systems and other components of the device that the agent is associated with.***

Pandya at 14:35-44.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1298. Pandya discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. To the extent this limitation is not explicitly or implicitly disclosed by Pandya, a person of ordinary skills in the art would have known that this requirement is obvious.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1299. Pandya discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. To the extent this limitation is not explicitly or implicitly disclosed by Pandya, a person of ordinary skills in the art would have known that this requirement is obvious.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*environment.*

1300. Pandya discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

As indicated in FIG. 4, each *control point 72 is typically associated with multiple agents 70, and the associated agents are referred to as being within a domain 76 of the particular control point. The control points coordinate and control the activity of the distributed agents within their domains.* In addition, the control points monitor the status of network resources, and share this information with management and support systems and with the agents.

Control points 72 and agents 70 may be flexibly deployed in a variety of configurations. For example, each agent may be associated with a primary control point and one or more backup control points that will assume primary control if necessary. Such a configuration is illustrated in FIG. 4, where control points 72 within the dashed lines function as primary connections, with the control point associated with server device 20 functioning as a backup connection for all of the depicted agents. *In addition, the invented management solution may be deployed so that one control point coordinates and controls the activity of a single domain, or of multiple domains.* Alternatively, one domain may be controlled and coordinated by the cooperative activity of multiple control points. In addition, agents may be configured to have embedded control point functionality, and may therefore operate without an associated control point entity.

Pandya at 7:3-26.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

1301. Pandya discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

In addition, *administrator module 134 stores system policies and/or components of policies, and provides policy data to various agent components as needed to implement and enforce the policies.* Administrator module 134 also includes support for interfacing the invented system with standardized network management protocols and platforms.

Pandya at 13:35-41.

1302. Further examples include Pandya at 20:13-22.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1303. Pandya discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

Referring now to FIGS. 13-16, both the agents and control points may be configured using configuration utility 106. Typically, configuration utility 106 is a platform-independent application that ***provides a graphical user interface for centrally managing configuration information for the control points and agents***. To configure the control points and agents, the configuration utility interface with administrator module 134 of agent 70 and with administrator module 168 of control point 72. Alternatively, configuration utility 106 may interface with administrator module 168 of control point 72, and the control point in turn may interface with administrator module 134 of agent 70.

Pandya at 20:39-50.

1304. Further examples include Pandya at 20:51-21:54; Pandya at Figs. 13, 14, 16.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

1305. Pandya discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1306. Pandya discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl.

26. For example:

The agents and control points interact to control and monitor network events, track operational and congestion status of network resources, select optimum targets for network requests, ***dynamically manage bandwidth usage***, and share information about network conditions with customers, users and IT personnel.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Pandya at 4:40-46.

1307. Further examples include Pandya at 16:28-17:35, 17:36-18:44, 4:40-45, 8:16-20; 14:54-67, 19:32-44; Pandya at Figs. 11A-D.

18. **Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

1308. Pandya discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

FIGS. 11A, 11B, 11C and 11D depict examples of various methods that may be implemented by traffic module 160 to dynamically allocate bandwidth. FIG. 11A depicts a process by which traffic module 160 determines whether any adjustments to bandwidth allocations AB are necessary...

Second, if there are any agents for which  $AB < CB$  and  $UB \cong AB$ , the allocation for those agents is modified, as seen in steps S6 and S10. The allocations for any such agent are typically increased....Third, if there are any agents reporting bandwidth usage UB that is less than their allocation AB, as determined at step S8, then the allocation AB for such an agent is reduced for the upcoming period to free up the unused bandwidth. Steps S4, S6 and S8 may be performed in any suitable order. Collectively, these three steps ensure that certain bandwidth allocations are modified, i.e. increased or reduced, if one or more of the following three conditions are true: (1)  $AB_{total} > CB_{total}$ , (2)  $AB < CB$  and  $UB \cong AB$  for any agent, or (3)  $UB < AB$  for any agent. If none of these are true, the allocations AB from the prior period are not adjusted. Traffic module 160 modifies allocations AB as necessary at step S10. ***After all necessary modifications are made, the control point communicates the new allocations to the agents for enforcement during the upcoming cycle.***

Pandya at 16:28-17:35.

1309. Further examples include Pandya at 17:36-18:44, 4:40-45, 8:16-20, 14:54-67, 19:32-44, 11:24-12:14; Pandya at FIGS. 11A-D.

19. **Claim 30**

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform a method comprising*

1310. Pandya discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

Software, systems and ***methods for managing a distributed network environment including a plurality of computers interconnected by a network link***, where at least some of the computers include a layered communications protocol stack for providing a data interface between an application program and the network link, the communications stack having a transport protocol layer for providing an end-to-end communications connection. The invention includes a control module and a plurality of agent modules, each agent being associated with one of the computers and adapted to dynamically monitor the associated computer at a data transmission point between an application program running on the computer and the transport protocol layer and repeatedly communicate with the control module in order to effect management of the distributed network system. The invented software, systems and methods may also include a messaging feature for providing users, IT personnel, or various management systems with informative messages concerning network conditions and network resources.

Pandya at Abstract.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1311. Pandya discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

1312. Pandya discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

1313. Pandya discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- e. *is able to clone itself;*

1314. Pandya discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1315. Pandya discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1316. Pandya discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1317. Pandya discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

1318. Pandya discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1319. Pandya discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network by replacing the assigned goal based on the optimal policy.*

1320. Pandya discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1321. Pandya discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1322. Pandya discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

1323. Pandya discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task;*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*and*

1324. Pandya discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1325. Pandya discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1326. Pandya discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1327. Pandya discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1328. I expect to testify that Pandya anticipates and/or renders obvious each Asserted Claim of the '730 Patent. A claim chart illustrating that Pandya discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-4.

**D. Anticipation by and/or Obviousness in View of United States Patent No.**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**6,012,152, issued to Douik, et al. on 1/4/2000.**

1329. As explained in detail below and in the chart attached as Ex. A-5, Douik anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

1330. Douik discloses the preamble of claim 1, "[a] method of managing a computer network, comprising." '730 Pat. at Cl. 1. For example:

***Network management*** means deploying and coordinating resources in order to plan, operate, administer, analyze, evaluate, design and expand communication networks to meet service level objectives at all times, at reasonable cost and with optimum capacity. Network management developments for mobile networks have almost the same objectives as for wired networks, the main objectives being to ensure good operation and service provisioning.

Douik at 2:41-45.

1331. Further examples include Douik at 3:10-12.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1332. Douik discloses this element of claim 1, "assigning a goal to a software [agent], wherein the software agent has its own runtime environment." '730 Pat. at Cl. 1. For example:

Thus, in one aspect, the present invention is a Software Fault Management (SFM) system for managing software faults in a managed mobile telecommunications network. The SFM system includes an Intelligent Management Information Base (I-MIB) comprising a Management Information Base (MIB) and a Knowledge Base (KB) having a functional model of the managed network. ***The SFM system also includes an intelligent multi-agent portion having a plurality of agents which process the Software faults utilizing information from the I-MIB.*** The intelligent multi-agent portion utilizes model-based reasoning to process the Software faults. The KB may include a trouble report/known faults (TR/KF) case base, and the intelligent multi-agent portion may utilize model-based reasoning in combination with case-based reasoning to process the Software faults. Fault management is both proactive and reactive.

Douik at 11:37-53.

1333. Further examples include Douik at 13:15-21, 22:34-48.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1334. To the extent that this limitation is not expressly disclosed by Douik, this Claim Limitation was obvious to a person of ordinary skill in the art because agents are software programs that would have needed a runtime environment to operate in.

c. *is able to communicate with other software agents in the computer network;*

1335. Douik discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

The [Software Fault Management (SFM)] is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. ***The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network***”

Douik at 13:15-21.

1336. Further examples include Douik at 22:34-48, 25:27-41; Douik at Fig. 1.

d. *is capable of perceiving its own state*

1337. Douik discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

The [Software Fault Management (SFM)] is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. ***The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.***

Douik at 13:15-21.

1338. Further examples include Douik at 22:34-48; Douik at Fig. 1.

e. *and is able to clone itself,*

1339. Douik discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example, agents clone themselves because all agents of the same class share the same attributes, behaviors, operations, notifications, and packages.

1340. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*All managed-objects that share the same attributes, behavior, operations, notifications and packages belong to the same managed-object class.* To provide for a convenient means of reusing definitions in the creation of a new object class, the OSI Structure of management information introduces the concept of inheritance. *A new object class can be defined by adding additional attributes, operations, or notifications to an existing managed-object class.* The new object class is referred to as a subclass of the old object class, and the old object class is referred to as a super-class of the new object class. All object classes ultimately derive from a unique object class referred to as top". This is the ultimate super-class, and the other object classes form an inheritance hierarchy with top as the root.

Douik at 17:52-65.

1341. Further examples include Douik at 33:34-43, 14:57-67; Douik at Fig. 1.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1342. Douik discloses this element of claim 1, "and wherein the goal is a programmatic expression of a predefined task for the software agent." '730 Pat. at Cl. 1. For example:

Thus, in one aspect, the present invention is a Software Fault Management (SFM) system for managing software faults in a managed mobile telecommunications network. The SFM system includes an Intelligent Management Information Base (I-MIB) comprising a Management Information Base (MIB) and a Knowledge Base (KB) having a functional model of the managed network. *The SFM system also includes an intelligent multi-agent portion having a plurality of agents which process the Software faults utilizing information from the I-MIB.* The intelligent multi-agent portion utilizes model-based reasoning to process the Software faults. The KB may include a trouble report/known faults (TR/KF) case base, and the intelligent multi-agent portion may utilize model-based reasoning in combination with case-based reasoning to process the Software faults. Fault management is both proactive and reactive.

Douik at 11:37-53.

1343. Further examples include Douik at 13:15-22, 22:34-48.

- g. *monitoring the computer network;*

1344. Douik discloses this element of claim 1, "monitoring the computer network." '730 Pat. at Cl. 1. For example:

The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. *The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Douik at 13:15-22.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1345. Douik discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

***The diagnosis agent 28 analyzes and tests the suspect software components against their modeled behaviors*** under test to verify the explanations supplied by the correlation agent. The diagnosis agent may execute the tests either automatically or wit [sic] the help of a human operator. The output of the verification process is a diagnosis 29 of the identity of the software component which has to be corrected, or if no explanation could be verified, a message to the correlation agent.

Douik at 15:32-40.

1346. Further examples include Douik at 37:25-42.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1347. Douik discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

***The diagnosis agent 28 analyzes and tests the suspect software components against their modeled behaviors*** under test to verify the explanations supplied by the correlation agent. The diagnosis agent may execute the tests either automatically or wit [sic] the help of a human operator. The output of the verification process is a diagnosis 29 of the identity of the software component which has to be corrected, or if no explanation could be verified, a message to the correlation agent.

Douik at 15:32-40.

1348. Further examples include Douik at 20:50-63, 21:12-21, 23:45-24:3, 29:13-26, 37:25-42.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1349. Douik discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

***The diagnosis agent 28 analyzes and tests the suspect software components against their modeled behaviors*** under test to verify the



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

explanations supplied by the correlation agent. The diagnosis agent may execute the tests either automatically or wit [sic] the help of a human operator. ***The output of the verification process is a diagnosis 29 of the identity of the software component which has to be corrected, or if no explanation could be verified, a message to the correlation agent.***

Douik at 15:32-40.

1350. Further examples include Douik at 20:51-63, 21:12-32, 23:45-24:3, 29:13-26, 37:25-42.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1351. Douik discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl.

1. For example:

The proactive monitoring of the managed cellular network occurs in a monitoring mode in which the [Software Fault Management (SFM)] system continually monitors, through dynamic polling, the state and behavior of critical resources in the cellular switching system. ***It analyzes performance and historical data and detects possible abnormal behaviors of what would eventually disturb the service in order to predict, and hence prevent, the occurrence of potential software faults.*** For example, based on selected performance data and statistics, the system may recognize a progressive degradation of Quality of Service (QoS). The proactive monitoring of the cellular network can also be used to manage such areas as digital quality service, software and hardware fault management, network monitoring, system characteristics and performances, and traffic monitoring. The proactive mode is initially effective for those faults that are well known (e.g., have a precise fault model, being part of well modeled fault scenarios, having intermediate symptoms, etc.) but also applies to new classes of faults. When polling indicates that a potential fault may occur, additional verifications are performed. Preventive measures are then automatically taken (if available), or a notification is sent to the system users if automated preventive measures are not available.

Douik at 13:36-58.

1352. Further examples include Douik at 22:34-4, 23:1-20, 29:13-26.

- 1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1353. Douik discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Management information for telecommunication systems is not always found at one single-logical or physical location. Normally, the management of a large network is distributed over various managers which manage (arbitrary) parts of the network. This means that the model of the overall network is cut into pieces and stored at different managers. In the area of [Software Fault Management (SFM)], several SFM Systems may cooperate, with each one being responsible for a different part of the overall model. ***Whenever managers need information beyond their model knowledge, they ask higher level managers which in turn have the right to request information from all subordinate managers.*** With the cooperation and the necessary interfaces between the model parts, boundaries between management domains...

Douik at 25:27-40.

1354. Further examples include Douik at Fig. 1.

1355. To the extent this limitation is not explicitly disclosed by Douik, it would have been obvious to a person of ordinary skill in the art. Douik contemplates software agents relying on code and functions from other agents, and discloses agents exchanging information between one another. It would have been obvious to a person of ordinary skill in the art at the time of invention to modify Douik so that an agent may request a policy from another entity when it requires additional functionality, or when another agent reports superior results.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1356. Douik discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

In addition to structural representations, the agents provide facilities to describe the functional behavior of the cellular switching system components. ***The behavior is normally described in the form of rules (e.g., if-then-rules) which are attached to the defined classes in the Knowledge Base 38.*** The acquisition and representation agents also enable users to interact with the system reasoning agents to test rule behavior and to perform simulations and inferences on the mobile switching system model 39 as represented in the I-MIB 36.

Douik at 21:12-21.

1357. Further examples include Douik at 11:64-12:19.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*about a network component by the software agent in performing the predefined task; and*

1358. Douik discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. ***The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.***

Douik at 13:15-21.

1359. Further examples include Douik at 20:64-21:11.

- b. *constructing a topological representation of the computer network from the information.*

1360. Douik discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

The [Software Fault Management (SFM)] system 10 of the present invention utilizes integrated intelligent agents to support users in acquiring and representing mobile telecommunications network knowledge. ***These agents allow the representation of network elements and their connectivity (e.g., the switch software blocks and their relationships depending on the mobile service logic) within the Knowledge Base 38. The representations may be graphical and correspond to the concepts of abstract classes and instances of the [Management Information Base (MIB)].*** The agents implement several object management operations (e.g., add, remove) and other transactions of knowledge within the knowledge base in order to keep the knowledge base consistent. Browsing facilities are also provided by the agents to cover all classes and instances in the knowledge base.

Douik at 20:65-21:11.

**4. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1361. Douik discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Model-based diagnostic techniques describe reasoning on the basis of quantitative or qualitative device models to diagnose failures. ***Quantitative models include simulations and numerical models.*** Qualitative models include structural, behavioral, and functional black box models.

Douik at 7:2-6.

1362. Further examples include Douik at 8:25-36.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1363. Douik discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

***Automated Fault Management***

There are several existing knowledge-based and artificial intelligence (AI) techniques that can be used for fault diagnosis. Five categories relevant to fault diagnosis are identified: fault-based techniques, model-based techniques, case-based reasoning techniques, machine learning for knowledge acquisition, and integrated diagnostic techniques. A description of the techniques and how they apply to diagnosis follows.

Douik at 6:31-39.

1364. Further examples include Douik at 20:65-21:11, 31:23-37.

1365. To the extent that this limitation is not expressly disclosed by Douik, this Claim Limitation was obvious to a person of ordinary skill in the art because Dijkstra’s Self Stabilizing Algorithm was a well-known algorithm for automated fault management of networks at the time.

6. ***Claim 7***

- a. *A computer network, comprising:*

1366. Douik discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

***Network management*** means deploying and coordinating resources in order to plan, operate, administer, analyze, evaluate, design and expand communication networks to meet service level objectives at all times, at reasonable cost and with optimum capacity. Network management developments for mobile networks have almost the same objectives as for wired networks, the main objectives being to ensure good operation and service provisioning.

Douik at 2:41-48.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1367. Further examples include Douik at Douik at 3:10-13, 8:60-9:3.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1368. Douik discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

1369. Douik discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

1370. Douik discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

1371. Douik discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

- f. *is able to clone itself;*

1372. Douik discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

- g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1373. Douik discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

***The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems. The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Douik at 13:15-21.

1374. Further examples include Douik at 32:5-18; Douik at Fig. 4.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1375. Douik discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

1376. Douik discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

1377. Douik discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1378. Douik discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1379. Douik discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1380. Douik discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1381. Douik discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

FIG. 1 is an overall functional block diagram illustrating the functional components of the SFM system 10 and interactions between the [Software Fault Management (SFM)] and human operators 21 through a Graphical User Interface (GUI) 22. ***To perform the complete SFM function, the communication between the key agents, event report management, correlation, diagnosis and repair has to be coordinated. For that purpose, a coordinator super-agent 23 is introduced to coordinate the overall SFM cycle.*** The coordinator super-agent also manages (creates instances, removes instances, etc.) the agents responsible for the different tasks involved in the SFM cycle. Functional models can exist at different levels and for different components of the system to be managed. Thus, in the overall SFM process, there may be several instances of the agents involved in the SFM cycle.

Douik at 14:57-67.

1382. Further examples include Douik at Fig. 1.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1383. Douik discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

FIG. 4 is a block diagram of a physical architecture in the preferred embodiment of the SFM system 10 of the present invention. ***The architecture is compliant with Telecommunication Management Network (TMN) principles and framework.*** There are four logical layers of the TMN architecture: Service Management, Network Management, Network Element Management, and Network Element Layer. A block is considered to be physical when it is implemented on independent physical equipment, and it communicates with other blocks through TMN interfaces. For this reason, most of the network elements are presented as single physical blocks. Internally, they are made of several independent functional blocks which may be distributed on different equipment.

Douik at 32:5-18.

1384. Further examples include Douik at Fig. 4.

1385. To the extent that this limitation is not expressly disclosed by Douik, this Claim Limitation was obvious to a person of ordinary skill in the art because Telecommunication Network Management principles includes the use of secure communications protocols.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

1386. Douik discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

***The [Software Fault Management (SFM)] system is an integrated collection of autonomous agents which support the software fault management of existing cellular telecommunications switching systems.*** The SFM agents, working on different network elements and/or on different aspects of the software fault management process cooperate in order to provide additional and more global information to assist in the diagnosis of problems in the cellular network.

Douik at 13:15-21.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

1387. Douik discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.”

’730 Pat. at Cl. 17. For example:

FIG. 1 is an overall functional block diagram illustrating the functional components of the [Software Fault Management (SFM)] system 10 and interactions between the SFM system and human operators 21 through a Graphical User Interface (GUI) 22. To perform the complete SFM function, the communication between the key agents, event report management, correlation, diagnosis and repair has to be coordinated. For that purpose, a coordinator super-agent 23 is introduced to coordinate the overall SFM cycle. The coordinator super-agent also manages (creates instances, removes instances, etc.) the agents responsible for the different tasks involved in the SFM cycle. Functional models can exist at different levels and for different components of the system to be managed. Thus, in the overall SFM process, there may be several instances of the agents involved in the SFM cycle.

Douik at 14:57-67.

1388. Further examples include Douik at Fig. 1.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1389. Douik discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

FIG. 1 is an overall functional block diagram illustrating the functional components of the [Software Fault Management (SFM)] system 10 and interactions between the SFM system and human operators 21 through a Graphical User Interface (GUI) 22. To perform the complete SFM function, the communication between the key agents, event report management, correlation, diagnosis and repair has to be coordinated. For that purpose, a coordinator super-agent 23 is introduced to coordinate the overall SFM cycle. ***The coordinator super-agent also manages (creates instances, removes instances, etc.) the agents responsible for the different tasks involved in the SFM cycle.*** Functional models can exist at different levels and for different components of the system to be managed. Thus, in the overall SFM process, there may be several instances of the agents involved in the SFM cycle”

Douik at 14:57-67.

1390. Further examples include Douik at Fig. 1.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**13. **Claim 19**

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

1391. Douik discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

FIG. 1 is an overall functional block diagram illustrating the functional components of the [Software Fault Management (SFM)] system 10 and interactions between the SFM system and human operators 21 through a Graphical User Interface (GUI) 22. To perform the complete SFM function, the communication between the key agents, event report management, correlation, diagnosis and repair has to be coordinated. For that purpose, a coordinator super-agent 23 is introduced to coordinate the overall SFM cycle. ***The coordinator super-agent also manages (creates instances, removes instances, etc.) the agents responsible for the different tasks involved in the SFM cycle.*** Functional models can exist at different levels and for different components of the system to be managed. Thus, in the overall SFM process, there may be several instances of the agents involved in the SFM cycle.

Douik at 14:57-67.

1392. Further examples include Douik at 18:51-61; Douik at Fig. 1.

14. **Claim 21**

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

1393. Douik discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

In another aspect, the present invention is a method of managing software faults in a managed mobile telecommunications network. ***The method begins by storing a Knowledge Base (KB) in an Intelligent Management Information Base (I-MIB), the KB including a functional model of the managed network.*** The method also includes the steps of storing a Management Information Base (MIB) in the I-MIB and processing the software faults with a plurality of agents in an intelligent multi-agent system utilizing information from the I-MIB.

Douik at 11:54-63.

1394. Further examples include Douik at 23:57-24:03.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1395. Douik discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

These system capabilities are achieved by automatically acquiring network and traffic data from the switches, storing the data and ***presenting value-added information via the GUI 22 to trouble shooters and engineers 21.*** Costly equipment down time is reduced by predicting the occurrence of faults before the client perceives trouble, based upon minimal performance criteria for each switch.

Douik at 29:61-67.

1396. Further examples include Douik at Fig. 1.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

1397. Douik discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1398. Douik discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The model analyzer sub-agent 34 performs deductions from a hypothetical explanation (i.e., context explanation). It utilizes only those rules which are appropriate in the context. The MIB 37 is queried for the state values of the involved managed objects, and the model analyzer sub-agent 34 determines if the context explanation is consistent. The coordinator sub-agent 32 invokes the model analyzer sub-agent with partial explanations, i.e., those which account for the symptoms incorporated to date. ***If the context is found to be inconsistent, no more rules are used, and the hypothetical explanation is removed from the search by the coordinator sub-agent.***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The model analyzer sub-agent 34 performs two main functions: rule generation and rule interpreting. Rule generation consists of taking the rules as written for the functional blocks (which refer to internal states, operational states, and intermediate states) and utilizing the connectivity information (signal transmission) to generate rules that explicitly refer to adjacent functional blocks. Rule generation is also performed by the symptoms analyzer sub-agent 35 for a similar purpose. Once a rule set has been generated it is saved so that it need not be generated again. The model analyzer sub-agent 34 then performs its rule interpreting by testing these rules and by passing the deductions together with their justifications to the deductions synthesizer sub-agent 33.

Douik at 23:45-24:3.

18. **Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

1399. Douik discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

The model analyzer sub-agent 34 performs deductions from a hypothetical explanation (i.e., context explanation). It utilizes only those rules which are appropriate in the context. The MIB 37 is queried for the state values of the involved managed objects, and the model analyzer sub-agent 34 determines if the context explanation is consistent. The coordinator sub-agent 32 invokes the model analyzer sub-agent with partial explanations, i.e., those which account for the symptoms incorporated to date. If the context is found to be inconsistent, no more rules are used, and the hypothetical explanation is removed from the search by the coordinator sub-agent.

The model analyzer sub-agent 34 performs two main functions: rule generation and rule interpreting. ***Rule generation consists of taking the rules as written for the functional blocks (which refer to internal states, operational states, and intermediate states) and utilizing the connectivity information (signal transmission) to generate rules that explicitly refer to adjacent functional blocks.*** Rule generation is also performed by the symptoms analyzer sub-agent 35 for a similar purpose. Once a rule set has been generated it is saved so that it need not be generated again. The model analyzer sub-agent 34 then performs its rule interpreting by testing these rules and by passing the deductions together with their justifications to the deductions synthesizer sub-agent 33.

Douik at 23:45-24:3.

19. **Claim 30**

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform a method comprising*

1400. Douik discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

***FIG. 4 is a block diagram of a physical architecture in the preferred embodiment of the SFM system 10 of the present invention.*** The architecture is compliant with Telecommunication Management Network (TMN) principles and framework. There are four logical layers of the TMN architecture: Service Management, Network Management, Network Element Management, and Network Element Layer. A block is considered to be physical when it is implemented on independent physical equipment, and it communicates with other blocks through TMN interfaces. For this reason, most of the network elements are presented as single physical blocks. Internally, they are made of several independent functional blocks which may be distributed on different equipment.

Douik at 32:5-18.

1401. Further examples include Douik at Fig. 4.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1402. Douik discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

1403. Douik discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

1404. Douik discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

1405. Douik discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1406. Douik discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1407. Douik discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1408. Douik discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

1409. Douik discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1410. Douik discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

1411. Douik discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

**20. Claim 31**

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1412. Douik discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

**21. Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1413. Douik discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

1414. Douik discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

1415. Douik discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1416. Douik discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1417. Douik discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1418. Douik discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1419. I expect to testify that Douik anticipates and/or renders obvious each Asserted Claim of the '730 Patent. A claim chart illustrating that Douik discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-5.

**E. Anticipation by and/or Obviousness in View of United States Patent No. 6,584,502, filed 6/29/1999 and issued to Natarajan, et al. on 6/24/2003.**

1420. As explained in detail below and in the chart attached as Ex. A-7, Natarajan anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

1421. Natarajan discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

As the use of computer networks proliferates, there exists an increasing need to improve computer network designs and implementations in order to facilitate the ***management, implementation, and modification of such networks***.

Natarajan at 2:11-15.

1422. Further examples include Natarajan at 2:17-24, 2:25-28, 6:7-7:11, 7:54-64.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1423. Natarajan discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example, Natarajan discloses a network element that communicates with a policy engine in connection with a monitor system and an ADMIN system, where the network element corresponds to the claimed “software agent.”

1424. Additionally, Natarajan discloses the policy engine in connection with the monitor system which communicates with the ADMIN system, where the combination of the policy engine and the monitor system may also correspond to the claimed “software agent.”

1425. For example:

Referring to FIG.16 , a frame relay virtual circuit is shown connecting user1 (1602) with user2 (1620). User1 communicates with a first router 1604 via ***link A***. In the example of FIG. 16, router 1604 may be managed by a first service provider (SP1). Router 1604 communicates with a frame relay cloud 1612 via ***link B*** and Switch 1612. The frame relay cloud 1610 is managed and maintained by a service provider such as AT&T. The frame relay cloud 1610 communicates with a second router 1614 via ***link C*** and switch 1613. The second router 1614 may be managed by a second service provider (SP2). User 1620 communicates with router 1614 via ***link D***.

Natarajan at 29:43-54.

1426. Further examples include Natarajan at 18:42-59, 32:39-46, 14:40-56, 11:31-34, 11:57-12:17, 14:56-67, 27:4-11, 30:36-45, 14:8-12, 14:20-27, 26:60-62; Natarajan at Figs. 2, 16.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *is able to communicate with other software agents in the computer network;*

1427. Natarajan discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

In the embodiment of FIG. 2, each of the network elements 204A and 204B may be, for example, a server or a router which ***communicate with each other*** via a wide area network (WAN) 210. The WAN may include a plurality of network elements (NE) 208A, 208B, which may include switches and/or other network elements for ***providing a communication link*** between element 204A and 204B.

Natarajan at 7:47-53.

1428. Further examples include Natarajan at 3:17-21, 19:44-48, 3:22-42, 2:53-67, 7:65-8:7.

- d. *is capable of perceiving its own state*

1429. Natarajan discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

The network element is preferably designed with ***built-in instrumentation which allows for the collection of relevant information*** which may be subsequently used to determine control actions to be applied to the network element.

Natarajan at 8:10-13.

1430. Further examples include Natarajan at 8:21-28, 2:32-34, 7:12-19, 10:41-57, 20:23-27, 8:39-47, 2:53-56, 31:47-32:17, 16:26-45, 18:19-38, 10:41-57, 20:23-27, 8:39-47, 33:10-53.

- e. *and is able to clone itself,*

1431. Natarajan discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

At 604, the remaining (i.e. non-control) elements of the network are initialized. In a specific embodiment, ***each element may be initialized using existing control parameters*** which reside in a local cache (e.g., 276A) or a configuration file (not shown) residing at the network element. Event registration is then initiated (606) for the various network elements (including network control elements)

Natarajan at 19:8-15.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1432. Further examples include Natarajan at 19:25-32, 15:41-46, 16:56-17:2, 30:11-17, 27:38-54, 30:5-9, 30:23-31, 30:36-45.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1433. Natarajan discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

The definition of “policy” varies based upon the perspective of the user. In the case of management and control of network elements, ***a policy is a corrective action*** which is used to restore the network element to a pre-determined state.

Natarajan at 14:15-19.

1434. Further examples include Natarajan at 15:8-11, 14:40-56, 7:24-32, 32:39-46, 15:12-18, 14:56-67, 32:27-38, 33:10-53.

- g. *monitoring the computer network;*

1435. Natarajan discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

For example, as shown in FIG. 3 ***various frame relay virtual circuit parameter values (e.g., CIR, EIR, dropped packets, etc.)*** associated with network element 304A are obtained and reported to data store 352.

Natarajan at 20:23-27.

1436. Further examples include Natarajan at 13:63-14:2, 17:3-5, 19:53-57, 3:22-42, 3:13-17, 8:10-13, 8:21-28, 2:32-34, 30:46-49, 32:39-46, 7:12-19, 8:39-47, 26:67-27:3, 27:12-26, 30:35-45, 31:35-47, 10:41-57, 33:5-11, 2:53-56; Natarajan at Fig. 8.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1437. Natarajan discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The *policy engine 254 is a decision-making (logical) component* of the feedback-based adaptive network of the present invention.

Natarajan at 14:5-7.

1438. Further examples include Natarajan at 14:30-32, 17:3-18:5, 31:5-19, 30:49-52, 32:39-46, 27:4-11, 28:25-38, 28:46-48, 28:59-29:3, 32:27-38.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1439. Natarajan discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

Additionally, the policy engine may be implemented with *neural networks* or with *other artificial intelligence technologies*.

Natarajan at 14:30-32.

1440. Further examples include Natarajan at 17:3-12, 32:39-46, 28:59-29:3, 7:37-43, 7:33-34, 31:5-19, 30:49-52.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1441. Natarajan discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

Presumably, the updating of the CIR parameter values in each of the selected network elements will affect the number of packets dropped on each of the links associated with these network elements. Thus, after the appropriate network elements have been updated or re-configured utilizing the updated CIR parameter values, *the adaptive feedback control process resumes at block 1706, whereupon each of the links A, B, C, D provides updated information* to the data store relating to the respective number of packets dropped at each link. *This updated information is then analyzed by the policy engine* to generate new CIR parameter values, (if necessary) to be implemented by the appropriate network elements. *This adaptive feedback-control process continues until the reported number of packets dropped by each of the respective links conforms with predetermined criteria.*

Natarajan at 31:5-19.

1442. Further examples include Natarajan at 30:49-52, 32:39-46, 14:30-32, 28:46-48, 28:59-29:3, 17:3-12, 32:27-38.

- k. *dynamically modifying the assigned goal of the software agent by*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*replacing the assigned goal based on the optimal policy*

1443. Natarajan discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

*The updated CIR data may then be used by various elements in the network to modify or affect each element's respective behavior or operation* (thereby affecting operation of the network). In this way, the network elements are able to automatically and dynamically adapt to changing network conditions.

Natarajan at 15:24-29.

1444. Further examples include Natarajan at 18:42-59, 24:54-62, 20:28-40, 28:46-48, 28:59-29:3, 16:26-45, 18:128-38, 29: 33-36; Natarajan at Figs. 10, 17, 18.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1445. Natarajan discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

An additional aspect of this embodiment provides that, *where the condition or status of the first device relates to an error* detected by the first device, *an event notification message relating to the error is reported* to at least one other network device.

Natarajan at 3:17-21.

1446. Further examples include Natarajan at 10:50-57, 21:12-16, 15:12-18, 23:62-24:9, 10:41-57, 21:12-16, 14:2-4, 7:12-19, 32:17-20, 33:10-53, 28:11-21; Natarajan at Figs. 9B, 13.

## 2. **Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1447. Natarajan discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

The definition of “policy” varies based upon the perspective of the user. In the case of management and control of network elements, *a policy is a corrective action* which is used to restore the network element to a pre-determined state.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Natarajan at 14:15-19.

1448. Further examples include Natarajan at 15:8-11, 14:40-56, 7:24-32, 32:39-46, 15:12-18, 14:56-67, 32:27-38, 33:10-53.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1449. Natarajan discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

The network element is preferably designed with ***built-in instrumentation which allows for the collection of relevant information*** which may be subsequently used to determine control actions to be applied to the network element.

Natarajan at 8:10-13.

1450. Further examples include Natarajan at 10:31-57, 19:53-57, 20:23-27, 3:22-42, 3:13-17, 8:21-28, 2:32-34, 30:46-49, 33:5-10, 2:53-56, 7:12-19, 8:39-47, 26:67-27:3, 27:12-26, 30:35-45, 7:54-64, 33:10-53.

- b. *constructing a topological representation of the computer network from the information.*

1451. Natarajan discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

In one embodiment, the network element may ***determine the appropriate data stores for reporting its operating information*** by consulting a configuration file residing at the network element. In a specific embodiment, an automated approach using a ***multicast mechanism for discovering the appropriate data stores*** may be implemented via additional software. The automated approach using multicast automates the data store selection process and may therefore be independent of the particular implementation of each data store.

Natarajan at 19:61-20:3.

1452. Further examples include Natarajan at 7:54-64; Natarajan at Figs. 2, 3, 4, 16.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*method.*

1453. Natarajan discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

A script node may be composed of multiple read nodes, write nodes, state nodes, **computation nodes**, etc. Script nodes compute the next State in the policy tree based upon the results of the computation. ***If the current node is a compute node (1114), the policy server performs the specified computation (1126).***

Natarajan at 15:62-67.

1454. Further examples include Natarajan at 17:3-18:5, 14:30-32, 32:27-38, 28:59-29:3.

5. ***Claim 6***

a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1455. Natarajan discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

Additionally, the policy engine may be implemented with **neural networks** or with **other artificial intelligence technologies**.

Natarajan at 14:30-32.

1456. Further examples include Natarajan at 28:59-29:3, 7:54-64.

6. ***Claim 7***

a. *A computer network, comprising:*

1457. Natarajan discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

It will be appreciated, however, that the dynamic feedback-based adaptive network technique of the present invention may be implemented in **any conventional data network** for providing adaptive and automatic feedback control of network elements within that network. Thus, for example, the technique of the present invention may be implemented in conventional LANs, WANs, MANs, internetworks, general purpose networks, packet Switched networks, circuit Switched networks, etc. Moreover, the technique of the present invention may be applied to any conventional data network.

Natarajan at 7:54-64.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*hardware*

1458. Natarajan discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

1459. Natarajan discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

1460. Natarajan discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

1461. Natarajan discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

1462. Natarajan discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1463. Natarajan discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

Generally, the dynamic feedback-based adaptive ***network element*** of the present invention ***may be implemented using*** software and/or ***hardware***. For example, it can be implemented in an operating system kernel, in a separate user process, in a library package bound into network applications, on a specially constructed machine, or on a network interface card.

Natarajan at 11:25-31.

1464. Further examples include Natarajan at 8:13-20, 14:8-12, 14:20-27, 8:52-9:2, 9:28-44, 9:55-10:2, 11:35-12:38, 12:46-59, 14:8-12, 14:20-27.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1465. Natarajan discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

1466. Natarajan discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

1467. Natarajan discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1468. Natarajan discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1469. Natarajan discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*an operational characteristic of the network.*

1470. Natarajan discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1471. Natarajan discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

In specific embodiments where the protocol for accessing data within data store 252 differs from the protocol for accessing data within data cache 276A, a ***translation mechanism*** may be provided. For example, where the data store is implemented as an ***LDAP server***, and the local cache is implemented as a ***file system***, a translation mechanism will be provided at the network element for ***converting LDAP calls into file system calls and vice versa***.

Natarajan at 9:20-28.

1472. Further examples include Natarajan at 3:22-42, 10:7-11, 10:16-21, 10:21-57, 10:41-57, 10:62-11:16, 13:32-36, 18:42-59, 20:6-19, 20:64-67, 21:1-10, 21:25-27, 21:50-22:11, 22:15-32, 23:39-51, 25:23-26, 22:42-44, 25:27-29, 26:3-10, 11:16-19, 28:59-29:3, 29:4-36; Natarajan at Figs. 9A, 9B, 15.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1473. Natarajan discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

In specific embodiments where the protocol for accessing data within data store 252 differs from the protocol for accessing data within data cache 276A, a ***translation mechanism*** may be provided. For example, where the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

data store is implemented as an **LDAP server**, and the local cache is implemented as a **file system**, a translation mechanism will be provided at the network element for **converting LDAP calls into file system calls and vice versa**.

Natarajan at 9:20-28.

1474. Further examples include Natarajan at 20:6-19, 23:39-51, 26:3-10, 19:38-41, 11:16-19.

10. **Claim 16**

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

1475. Natarajan discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

In a specific embodiment of this invention, the technique of the present invention is implemented in software such as an **operating system or in an application running on an operating system**.

Natarajan at 11:31-34.

1476. Further examples include Natarajan at 11:57-12:17, 14:8-12, 14:20-27, 26:61-62, 27:4-11.

11. **Claim 17**

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1477. Natarajan discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

According to a specific embodiment, the event handler 274A may initially **consult a local configuration file (not shown) in order to determine which events the network element is to register for at the event server 270**. The configuration file may be programmed using a command line interface terminal 206A. Alternatively, the event handler may be statically pre-configured to automatically register for specified events upon initialization. Thus, at 702, the event handler 274A determines the particular events for which network element 204A is to be registered at the event server 270.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Natarajan at 19:25-35.

1478. Further examples include Natarajan at 11:31-34, 16:59-17:2.

12. **Claim 18**

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1479. Natarajan discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

At 604, the remaining (i.e. non-control) elements of the network are initialized. In a specific embodiment, ***each element may be initialized using existing control parameters*** which reside in a local cache (e.g., 276A) or a configuration file (not shown) residing at the network element. Event registration is then initiated (606) for the various network elements (including network control elements).

Natarajan at 19:8-15.

1480. Further examples include Natarajan at 19:25-32, 15:41-46, 16:56-17:2, 30:11-17, 27:38-54, 30:23-31, 30:36-45, 11:31-34, 16:1-6, 16:24-26, 18:6-12, 19:25-32, 27:38-54, 16:56-17:2; Natarajan at Fig. 11.

13. **Claim 19**

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

1481. Natarajan discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730

Pat. at Cl. 19. For example:

In specific embodiments where the protocol for accessing data within data store 252 differs from the protocol for accessing data within data cache 276A, a ***translation mechanism*** may be provided. For example, where the data store is implemented as an ***LDAP server***, and the local cache is implemented as a ***file system***, a translation mechanism will be provided at the network element for ***converting LDAP calls into file system calls and vice versa***.

Natarajan at 9:20-28

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1482. Further examples include Natarajan at 20:6-19, 23:39-51, 26:3-10, 19:38-41, 11:16-19, 19:25-32, 15:41-46, 16:56-17:2, 27:38-54, 30:23-31, 11:31-34, 11:35-12:38, 12:46-59, 14:8-12, 14:20-27.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

1483. Natarajan discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

In a specific embodiment, the policies which form part of the policy engine may be stored in a ***policy library***. Each policy may be implemented as a decision tree comprised of nodes, wherein at each node an action may be performed if specified criteria is satisfied. The policy may be configured as an application or program which runs on top of the operating system of the policy server. The policy engine traverses the decision trees of the various policies.

Natarajan at 15:1-8.

1484. Further examples include Natarajan at 14:8-14, 7:19-24.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1485. Natarajan discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

Generally, the dynamic feedback-based adaptive ***network element*** of the present invention ***may be implemented using software and/or hardware***. For example, it can be implemented in an ***operating system kernel, in a separate user process, in a library package bound into network applications, on a Specially constructed machine, or on a network interface card***.

Natarajan at 11:25-31.

1486. Further examples include Natarajan at 14:20-27, 11:35-12:38, 12:46-59, 33:54-61, 32:33-27, 28:25-38, 29:11-27.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*a Dijkstra Self Stabilization Algorithm.*

1487. Natarajan discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. **Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1488. Natarajan discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The policy engine may then generate ***new or updated control information or parameters*** for affecting the operation of selected elements in the network.

Natarajan at 7:34-37.

1489. Further examples include Natarajan at 31:19-34, 30:49-52, 15:24-29, 18:42-59, 24:54-62, 20:28-40, 28:46-48, 28:59-29:3, 16:26-45, 18:28-38, 29: 33-36; Natarajan Figs. 10, 17, 18.

18. **Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

1490. Natarajan discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

For example, ***the network element may execute a procedure*** for dynamically determining upper and lower bounds for a respective control parameters based upon its current operating condition and/or performance. The upper and lower bounds for these parameters may then be used to validate the control information retrieved from the data store. ***If the retrieved values from the data store are determined to be invalid, the network element may simply ignore the updated parameter data and continue to behave using its current control parameter information.*** Additionally, where the values in the data store are determined to be invalid, the network element may report the error or problem to the event server

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

270, which may then forward the error information to other appropriate control elements for logging or responding to the error.

Natarajan at 23:62-24:9.

1491. Further examples include Natarajan at 17:3-28, 28:57-58, 31:35-32:13, 14:30-32, 14:40-56.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

1492. Natarajan discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

Because such information and program instructions may be employed to implement the systems/methods described herein, ***the present invention relates to machine readable media*** that include program instructions, operating information, etc. for performing various operations described herein. Examples of machine-readable media include, but are not limited to, magnetic media Such as hard disks, floppy disks, and magnetic tape, optical media Such as CD-ROM disks; magneto-optical media such as floptical disks, and hardware devices that are specially configured to store and perform program instructions, such as read-only memory devices (ROM) and random access memory (RAM). The invention may also be embodied in a carrier wave travelling over an appropriate medium such as airwaves, optical lines, electric lines, etc. Examples of program instructions include both machine code, such as produced by a compiler, and files containing higher level code that may be executed by the computer using an interpreter.

Natarajan at 12:60-13:11.

1493. Further examples include Natarajan at 11:25-31, 14:20-29, 8:13-20.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1494. Natarajan discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

1495. Natarajan discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

d. *is capable of perceiving its own state; and*

1496. Natarajan discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself;*

1497. Natarajan discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1498. Natarajan discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1499. Natarajan discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1500. Natarajan discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

1501. Natarajan discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1502. Natarajan discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

1503. Natarajan discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1504. Natarajan discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1505. Natarajan discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

1506. Natarajan discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

1507. Natarajan discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1508. Natarajan discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1509. Natarajan discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1510. Natarajan discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1511. I expect to testify that Natarajan anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Natarajan discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-7.

**F. Anticipation by and/or Obviousness in View of United States Patent No. 5,655,081, issued to Bonnell, et al. on 8/5/1997.**

1512. As explained in detail below and in the chart attached as Ex. A-15, Bonnell anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

1513. Bonnell discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. ***The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network.*** An agent software system is installed on and runs on each of the server computer systems in the network. Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

- b. *assigning a goal to a software [agent], wherein the software agent*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*has its own runtime environment*

1514. Bonnell discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is installed on and runs on each of the server computer systems in the network. ***Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.***

Bonnell at 6:61-7:8.

1515. Further examples include Bonnell at Figs. 9, 11.

c. *is able to communicate with other software agents in the computer network;*

1516. Bonnell discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

FIG. 3 illustrates the main components of the agent software system 36 shown in FIG. 1. ***Communications module 62 coordinates message communications to and from other computers,*** such as network management computer system 10, and parses the information contained in such messages.

Bonnell at 3:10-15.

1517. Further examples include Bonnell at 7:51-56; Bonnell at Fig. 3.

1518. To the extent that this limitation is not expressly disclosed by Bonnell, this limitation would have been obvious to a person of ordinary skill in the art because the coordination of multiple agents implies an ability to communicate with other agents.

d. *is capable of perceiving its own state*

1519. Bonnell discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. *An agent software system is installed on and runs on each of the server computer systems in the network. Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.*

Bonnell at 6:61-7:8.

e. *and is able to clone itself,*

1520. Bonnell discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example, an agent loads a set of default policies when it is initialized. Thus, the agent clones itself every time it is re-initialized:

***FIG. 16 is a flow diagram illustrating a preferred procedure for initializing agent 202.*** In step 240, knowledge modules are stored at the site of agent 202 in non-volatile memory, such as in storage device 26. In step 242, agent software 202 begins executing. In step 244, agent software 202 checks a configuration file, also preferably stored on storage device 26, indicating which resources or applications on the server are to be monitored always, regardless of whether or not a console has registered interest in the application (such applications are hereinafter called "default resources"). ***In step 246, agent 202 loads, from storage device 26 into knowledge database 75, only the knowledge modules that correspond to the default applications. In step 248, agent 202 initializes run queue 71 so that the default applications will be monitored periodically according to the information and instructions contained in the loaded knowledge modules.***

Bonnell at 10:42-58.

1521. Further examples include Bonnell at Fig. 16.

1522. As an additional example, a collector agent can send a set of policies to another agent, asking the other agent to monitor some aspect of the network and to send the findings back to the collector. Every time the collector transfers a set of policies from itself to another agent, it is cloning a part of itself.

#### Multi-Tiered Problem Management

FIG. 27A and 27B comprise a block diagram illustrating an alternative embodiment of the invention, configured to yield multi-tiered problem monitoring and management capabilities in a large-scale enterprise. In such a configuration, the network is effectively divided into two or more tiers, such as tiers 362 and 364. It is to be understood that agent 356 in the drawing

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

represents a multitude of other similar agents operating in tier 362 in the network. ***Collector 358 may be a single intermediary agent***, or it may be viewed for purposes of the illustration as representing a number of such intermediate agents operating between tiers of a network, such as between tiers 362 and 364. The effect of the configuration is to reduce event-related network traffic on the higher tiers of the network, and also to remove some of the load of event management from higher level consoles. ***In operation, collector 358 registers with agents 356 as an SNMP manager and therefore begins to receive notification of SNMP traps.***

Bonnell at 14:49-67.

1523. Further examples include Bonnell at Figs. 27a, 27b.

1524. To the extent that this limitation is not expressly disclosed by Bonnell, this limitation would have been obvious to a person of ordinary skill in the art because Bonnell discloses initializing new agents and using a multitude of agents to monitor a network. It would have been obvious to have the agents clone themselves, for example, when the size of the network grows.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1525. Bonnell discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is installed on and runs on each of the server computer systems in the network. ***Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.***

Bonnell at 6:61-7:8.

1526. Further examples include Bonnell at Figs. 9, 11.

g. *monitoring the computer network;*

1527. Bonnell discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is installed on and runs on each of the server computer systems in the network. ***Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted***

Bonnell at 6:61-7:8.

1528. Further examples include Bonnell at Figs. 9, 11.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1529. Bonnell discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1530. Bonnell discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1531. Bonnell discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1532. Bonnell discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

***In steps 304 and 306, the agent determines which knowledge modules will be required in the agent’s knowledge database 75 in order to service the requests of the registered console. The agent then proceeds to load the requisite knowledge modules. However, the agent will not re-load***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

knowledge modules that are already present in knowledge database 75, thus avoiding redundancy. ***This process results in dynamic loading of knowledge modules***, which is illustrated schematically in FIG. 20. In FIG. 20, it can be seen by way of example that, according to the registration information received in the registration process of FIG. 19, console 1 is interested in applications A and C, console 2 is interested in applications A and B, and console "n" is interested only in application D. Thus, only four knowledge modules are needed in knowledge database 75, knowledge modules corresponding to applications A, B, C and D. Therefore, only those knowledge modules are loaded. (Of course, other knowledge modules may be present in knowledge modules 75 as well, such as those associated with "default" applications.)

Bonnell at 12:32-51.

1533. Further examples include Bonnell at Figs. 8, 19, 20.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1534. Bonnell discloses this element of claim 1, "wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task." '730 Pat. at Cl. 1. For example, the agents can request further policy from the agent's knowledge database:

***In steps 304 and 306, the agent determines which knowledge modules will be required in the agent's knowledge database 75 in order to service the requests of the registered console. The agent then proceeds to load the requisite knowledge modules.*** However, the agent will not re-load knowledge modules that are already present in knowledge database 75, thus avoiding redundancy. ***This process results in dynamic loading of knowledge modules***, which is illustrated schematically in FIG. 20. In FIG. 20, it can be seen by way of example that, according to the registration information received in the registration process of FIG. 19, console 1 is interested in applications A and C, console 2 is interested in applications A and B, and console "n" is interested only in application D. Thus, only four knowledge modules are needed in knowledge database 75, knowledge modules corresponding to applications A, B, C and D. Therefore, only those knowledge modules are loaded. (Of course, other knowledge modules may be present in knowledge modules 75 as well, such as those associated with "default" applications.)

Bonnell at 12:32-51.

1535. Further examples include Bonnell at Figs. 8, 19, 20.

1536. To the extent this limitation is not explicitly disclosed by Bonnell, it would have been obvious to a person of ordinary skill in the art. Bonnell contemplates software agents relying



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

on code and functions from other agents, and discloses agents exchanging messages between one another. It would have been obvious to a person of ordinary skill in the art at the time of invention to modify Bonnell so that an agent may request a policy from another entity (agent, collector agent, network manager, etc.) when, for example, it requires additional functionality, or when another agent reports superior results.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1537. Bonnell discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is installed on and runs on each of the server computer systems in the network. ***Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.***

Bonnell at 6:61-7:8.

1538. Further examples include Bonnell at Figs. 9, 11.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1539. Bonnell discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. An agent software system is



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

installed on and runs on each of the server computer systems in the network. *Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted*

Bonnell at 6:61-7:8.

1540. Further examples include Bonnell at Figs. 9, 11.

- b. *constructing a topological representation of the computer network from the information.*

1541. Bonnell discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1542. Bonnell discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1543. Bonnell discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6.

6. ***Claim 7***

- a. *A computer network, comprising:*

1544. Bonnell discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. *An agent software system is installed on and runs on each of the server computer systems in the network.* Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

1545. Further examples include Bonnell at Fig. 11.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1546. Bonnell discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

1547. Bonnell discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

1548. Bonnell discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

1549. Bonnell discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

- f. *is able to clone itself;*

1550. Bonnell discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

- g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1551. Bonnell discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

It is another object of the present invention to provide an agent system for use in an enterprise management system wherein *the agent system utilizes*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the memory and CPU resources of a server computer system in an efficient manner*, regardless of the number of console systems that are monitoring the resources on the server.

Bonnell at 6:28-33.

1552. Further examples include Bonnell at 6:61-7:8; Bonnell at Fig. 11.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1553. Bonnell discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

1554. Bonnell discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

1555. Bonnell discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1556. Bonnell discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

1557. Bonnell discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1558. Bonnell discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1559. Bonnell discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. ***The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network.*** An agent software system is installed on and runs on each of the server computer systems in the network. Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted. ***Each agent is also able to carry on a dialog of communication with manager software systems via the network,*** so that the consoles on the network management computer systems can provide a continuously updated display representing all resources and applications present throughout the network as well as the state of each such resource or application.

Bonnell at 6:61-7:14.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1560. Further examples include Bonnell at 2:67-3:2; Bonnell at Fig. 13.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1561. Bonnell discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

1562. Bonnell discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. ***An agent software system is installed on and runs on each of the server computer systems in the network.*** Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

1563. Further examples include Bonnell at Fig. 11.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1564. Bonnell discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. ***An agent software system is installed on and runs on each of the server computer systems in the network.*** Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

1565. Further examples include Bonnell at Fig. 11.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1566. Bonnell discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. ***An agent software system is installed on and runs on each of the server computer systems in the network.*** Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

1567. Further examples include Bonnell at 10:42-58; Bonnell at Figs. 11, 16.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*environment.*

1568. Bonnell discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

At least one manager software system is installed on and runs on at least one of the networked computer systems designated as a network management computer system. The network management computer systems act as consoles for monitoring and managing resources present on server computer systems in the network. ***An agent software system is installed on and runs on each of the server computer systems in the network.*** Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted.

Bonnell at 6:61-7:8.

1569. Further examples include Bonnell at Fig. 11.

1570. Additionally, it is my opinion that each agent runtime environment must necessarily store a list of running processes.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

1571. Bonnell discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

***Knowledge database manager 74 creates and maintains a database 75, either in RAM or on a storage device such as a hard disk, containing knowledge received via messages from manager software system 34.*** The knowledge maintained in agent's database 75 differs from the knowledge contained in manager's database 47, however, in that agent's database 75 typically does not contain information pertinent to the display of information on the manager's console.

Bonnell at 3:36-44.

1572. Further examples include Bonnell at 10:42-58; Bonnell at Figs. 12, 16.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism includes the graphical user interface.*

1573. Bonnell discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

Object database manager 48 creates and maintains a database 49 representing all of the resources and applications (collectively, “objects”) present on the computer network, as well as information pertaining to the state of those objects, in a form that will be readily useable by a graphical user interface module 50. Databases 47 and 49 may be stored in RAM or on a storage device such as a hard disk. ***Graphical user interface 50 is responsible for communicating with display driver software in order to present visual representations of objects on the display of network management computer system 10.*** Such representations typically take the form of icons for objects. Also, graphical user interface module 50 coordinates the representation of pop-up windows for command menus and the display of requested or monitored data.

Bonnell at 2:36-51.

1574. Further examples include Bonnell at Fig. 13.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

1575. Bonnell discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1576. Bonnell discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl.

26. For example: Bonnell at Fig. 8 illustrates this requirement.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

1577. Bonnell discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

Each respective agent software system carries out tasks on the computer system in which it is installed such as discovering which resources and applications are present on the computer system, monitoring particular aspects of the resources and applications present on the computer system, and executing recovery actions automatically when such actions are warranted. ***The agents are capable of intelligent, autonomous operation.*** Knowledge modules are stored in a non-volatile storage device at the site of each agent software system and are loaded and unloaded into server memory dynamically as consoles register and de-register with the agents.

Bonnell at Abstract.

1578. Further examples include Bonnell at Fig. 8.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

1579. Bonnell discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

It is another object of the present invention to provide an agent system for use in an enterprise management system wherein ***the agent system utilizes the memory and CPU resources of a server computer system in an efficient manner***, regardless of the number of console systems that are monitoring the resources on the server.

Bonnell at 6:28-33.

1580. Further examples include Bonnell at 6:61-7:8; Bonnell at Fig. 11.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1581. Bonnell discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

1582. Bonnell discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

d. *is capable of perceiving its own state; and*

1583. Bonnell discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself;*

1584. Bonnell discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1585. Bonnell discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1586. Bonnell discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1587. Bonnell discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

1588. Bonnell discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1589. Bonnell discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

1590. Bonnell discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1591. Bonnell discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1592. Bonnell discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

1593. Bonnell discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

1594. Bonnell discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1595. Bonnell discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1596. Bonnell discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1597. Bonnell discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1598. I expect to testify that Bonnell anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Bonnell discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-15.

**G. Anticipation by and/or Obviousness in View of United States Patent No. 6,122,664, issued to Boukobza, et al. on 9/19/2000.**

1599. As explained in detail below and in the chart attached as Ex. A-16, Boukobza anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

1600. Boukobza discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

The present invention relates to ***a process for monitoring*** a plurality of object types of a plurality of nodes (N1, N2, ... , Nn) comprising a management node (MN) in ***an information system***.

Boukobza at Abstract

1601. Further examples include Boukobza at 1:11-13, 34:43-51, 2: 21-38, 1:57-64, 4:36-42; Boukobza at Fig. 1.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1602. Boukobza discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

In this respect, the monitoring process mentioned in the preamble is noteworthy in ***that the monitoring is configured and then distributed in a***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*filtered way from the management node to autonomous agents, an autonomous agent being installed in each node to be monitored* in order, by providing intertype correlation, either to monitor as near as possible the different object types or all of the object of a domain called a global object, defined generically, Boukobza at to be displayed through the graphical interface of the management node, each agent comprising a plurality of specific modules specific to the different object types or to a particular domain, each specific module measuring static and dynamic parameters specific to the object type it monitors and collecting these measurements, testing conditions on these parameters relative to predefined thresholds and possibly initiating actions associated with these tested conditions, which parameters, conditions and actions are modifiable by the user of the management node.

Boukobza at 2:21-38.

1603. Further examples include Boukobza at 2:21-38, 2:40-46, 3:30-39, 4:63-5:9, 5:13-18, 5: 34-42; Boukobza at Fig. 1.

c. *is able to communicate with other software agents in the computer network;*

1604. Boukobza discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

The sole FIGURE represents, in highly schematic fashion, an exemplary application of the process according to the invention to the ***intercommunication*** between a management node and a node to be monitored.

Boukobza at 4:29-32.

1605. Further examples include Boukobza at 21:42-56, 4:39-42; Boukobza at Fig. 1.

d. *is capable of perceiving its own state*

1606. Boukobza discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

Each autonomous agent installed in each node (or machine), *in*

***addition to the parameter measurements it performs, the conditions it evaluates (in this node), the actions (reconfiguration, correction, alert) associated with these conditions it initiates or the operations it performs later,*** feeds back to the management node the information to be displayed, such as for example, the change of state of the objects, the parameter values to be displayed in curve form, etc.

Boukobza at 3:30-39.

1607. Further examples include Boukobza at 4:5-15, 27:11-19, 6:30-35.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *and is able to clone itself,*

1608. Boukobza discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

Each agent is automatically "reactivable", which means that when an agent becomes active in a node, it begins by executing the command "mkitab" in order to place itself in the initialization table "inittab" so that it can be reactivated if the process dies or if the node malfunctions and is then reactivated. When an agent is deactivated, it executes the command "rmitab" so that it will not be reactivated, in which case the management node no longer recognizes it.

Boukobza at 27:11-19.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1609. Boukobza discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

During the configuration of the monitoring, the user of the management node describes the objects to be monitored [systems (UNIX, etc.), applications (Tuxedo, etc.), instances (Oracle, etc.), server (Informix, etc.), etc.] and specifies modifications relative to the default choice of the specific modules (for example, modification of the measurement period of a parameter, suppression of a parameter, addition of a parameter or a condition). For each new parameter to be measured, he describes the measurement command, indicates whether or not he desires to have the measurement displayed (in the form of a curve), specifies the conditions which will trigger an action (a sequence of operations) The action can consist of displaying the "down" status of an object (an Oracle instance, for example) using a function supplied by the product or of performing a test for correlation with a piece of system information (cpu utilization rate, for example) or Tuxedo information or information defined by the user.

Boukobza at 5:63-6:14.

1610. Further examples include Boukobza at 13:48-14:26, 6:36-37, 8:44-66.

g. *monitoring the computer network;*

1611. Boukobza discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

Finally, relative to specific module "system", it must be possible, among other things, *to monitor and measure the cpu time, the disk space, the inputs/outputs, the storage, the exchange rate, the number of users, the pagination, the network*, etc. Thus it is possible, for example, to measure



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the cpu utilization per second, with default display, or the input/output rate per second, or to identify the processors which are the largest users of cpu and collect them in order to perform an autonomous analysis ("offline", tbc).

Boukobza at 34:43-51.

1612. Further examples include Boukobza at Abstract, 1:11-13, 2:21-38, 2:39-52, 35:25-53; Boukobza at Fig. 1.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1613. Boukobza discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

In this respect, the monitoring process mentioned in the preamble is noteworthy in that the monitoring is configured and then distributed in a filtered way from the management node to autonomous agents, an autonomous agent being installed in each node to be monitored in order, by providing intertype correlation, either to monitor as near as possible the different object types or all of the object of a domain called a global object, defined generically, or to feed back information to be displayed through the graphical interface of the management node, each agent comprising a plurality of specific modules specific to the different object types or to a particular domain, each specific module measuring static and dynamic parameters specific to the object type it monitors and collecting these measurements, *testing conditions on these parameters* relative to predefined thresholds and possibly *initiating actions associated with these tested conditions*, which parameters, conditions and actions are modifiable by the user of the management node.

Boukobza at 2:20-38.

1614. Further examples include Boukobza at 2:46-52, 8:50-67.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1615. Boukobza discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

In this way, in order to ensure effective monitoring of the applications running in the plurality of nodes, the process applied in this case makes it

possible to measure specific parameters of each application, to *test conditions on these parameters* relative to thresholds, and *then to execute an action in order to warn of a problem, to reconfigure or to correct.*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Boukobza at 2:47-52.

1616. Further examples include Boukobza at 20:44-56; Boukobza at Fig. 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1617. Boukobza discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

The operation of the generic management agent occurs in the following way: reading of its configuration file. Syntactic and semantic analysis ("MOD IF \_PARAM" of a non-existent parameter, etc.) merging of the configuration file supplied by the administrator and the configuration file supplied by the specific modules. sending of the resulting configuration file to each agent (*filtering relative to the objects to be monitored in the machine and taking into account the problems* inherent to the malfunctioning nodes or to the nodes provided for backing up the data of a malfunctioning node in another node).

Boukobza at 21:42-55.

1618. Further examples include Boukobza at 8:50-67, 2:47-52.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1619. Boukobza discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

The operation of the generic management agent occurs in the following way: reading of its configuration file. Syntactic and semantic analysis ("MOD IF \_PARAM" of a non-existent parameter, etc.) merging of the configuration file supplied by the administrator and the configuration file supplied by the specific modules. sending of the resulting configuration file to each agent (*filtering relative to the objects to be monitored in the machine and taking into account the problems* inherent to the malfunctioning nodes or to the nodes provided for backing up the data of a malfunctioning node in another node).

Boukobza at 21:42-55.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

1620. Boukobza discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

Moreover, *a predefined or external action can request the value of a parameter of an object monitored by a node other than the node in which the action is executed.* In this case, one of the basic functions BF requesting the value *will transfer this request to the management node, which will then direct it to the appropriate node*, and the value will be returned in the reverse direction.

Boukobza at 5:40-46.

1621. Further examples include Boukobza at 14:27-63.

2. **Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1622. Boukobza discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

SCANLOG: for each object *the administrator can designate the "log" files* SL to be scanned and the "critical" errors to be searched for. If the option SCANLOG does not exist, nothing is done in the "log" files. If the "log" file LOG FILE is not described (the files to be scanned are not specified), each specific module has a predefined rule for constructing the list of files to be scanned.

Boukobza at 8:44-50.

1623. Further examples include Boukobza at 21:42-55, 6:36-38, 2:21-38, 4:63-5:8, 21:6-13.

3. **Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1624. Boukobza discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The action can be predefined or supplied by the administrator. It can include: *a request to display the status* ("ok", "down", "warning") of one or more "objects to be monitored" and the refreshing of the icon according to the status to be displayed. An "object to be monitored" is always displayed in the form of an icon.

Boukobza at 14:40-46.

1625. Further examples include Boukobza at 9:11-19, 9:23-31, 2:20-38.

- b. *constructing a topological representation of the computer network from the information.*

1626. Boukobza discloses this element of claim 3, "constructing a topological representation of the computer network from the information." '730 Pat. at Cl. 3. For example:

Preferably, during the monitoring, the status of the applications as well as the parameter curves selected by the administrator for their importance are displayed, making it possible to verify the proper functioning of the object or objects under supervision. Moreover, if the operator or the administrator desires, *in a punctual but interactive way, to view certain more detailed pieces of information*, a certain number of possibilities are available to him, such as the display of other measurement curves or the ability to "zoom" on measurements or parts of curves.

Boukobza at 3:2-11.

1627. Further examples include Boukobza at 17:5-29, 3:30-39.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1628. Boukobza discloses claim 4, "[t]he method of claim 1, wherein the modeling uses a numerical method." '730 Pat. at Cl. 4. For example:

The autonomous agent according to the idea of the invention is chiefly composed of a generic agent and of a plurality of modules specific to each type of object: Oracle, Tuxedo, system, DPG, FSX, SAP, etc. The global operation consists of *measuring* parameters, if necessary storing them in a "trace" file and allowing their display through the interface GUI, of evaluating the single or multiple conditions and of executing the action linked to the true condition, for all the objects described in the configuration file.

Boukobza at 21:6-14.

1629. Further examples include Boukobza at 3:25-29.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1630. Boukobza discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example: Boukobza at Figure 1 illustrate this claim.

6. ***Claim 7***

- a. *A computer network, comprising:*

1631. Boukobza discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

As seen above, the process according to the invention advantageously makes it possible to monitor n machines, that is n nodes, N1, N2, ... , Nn, from a management node MN. For the intercommunication to which the invention relates, the management node MN chiefly comprises a certain number of components, including the graphical user interface GUI and the configuration file CF.

Boukobza at 4:36-42.

1632. Further examples include Boukobza at Abstract, 1:57-64, 2: 21-38, 1:11-13; Boukobza at Fig. 1.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1633. Boukobza discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

1634. Boukobza discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

1635. Boukobza discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

1636. Boukobza discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

1637. Boukobza discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1638. Boukobza discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

A managed object, in this data processing environment, is a representation of a resource such as a ***machine***, a file, a peripheral, a user, an application, etc.

Boukobza at 1:33-35.

1639. Further examples include Boukobza at 3:30-39; Boukobza at Fig. 1.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1640. Boukobza discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*of a network component*

1641. Boukobza discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the prediction;*

1642. Boukobza discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1643. Boukobza discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1644. Boukobza discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1645. Boukobza discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**8. Claim 11**

a. *The computer network of claim 7, wherein the network control*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism comprises a communications mechanism to facilitate communications with agents.*

1646. Boukobza discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

The sole FIGURE represents, in highly schematic fashion, an exemplary application of the process according to the invention to the **intercommunication** between a management node and a node to be monitored.

Boukobza at 4:29-32.

1647. Further examples include Boukobza at 2:20-38; Boukobza at Fig. 1.

9. **Claim 12**

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1648. Boukobza discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

In this respect, the monitoring process mentioned in the preamble is noteworthy in that the monitoring is configured and then **distributed in a filtered way** from the management node to autonomous agents, an autonomous agent being installed in each node to be monitored in order, by providing **intertype correlation**, either to monitor as near as possible the different object types or all of the object of a domain called a global object, defined generically, or to feed back information to be displayed through the graphical interface of the management node, each agent comprising a plurality of specific modules specific to the different object types or to a particular domain, each specific module measuring static and dynamic parameters specific to the object type it monitors and collecting these measurements, testing conditions on these parameters relative to predefined thresholds and possibly initiating actions associated with these tested conditions, which parameters, conditions and actions are modifiable by the user of the management node.

Boukobza at 2:20-38.

10. **Claim 16**

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*for a particular network component.*

1649. Boukobza discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

In this way, thanks to the idea of the invention, the management node has excellent global visibility since the present process, once the monitoring has been configured in a filtered way, (which means that ***a configuration is specific to an object and its environment*** and that therefore the same object can be configured differently depending on its situation and its distribution), makes it possible to monitor a plurality of object types, to give them unified functions, and also to correlate, among these functions, measurements on different types of objects.

Boukobza at 3:40-49.

1650. Further examples include Boukobza at 3:25-29, 2:20-38, 3:29-39.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1651. Boukobza discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

***The configuration file CF contains all of the configurations of the objects*** with the description of these objects, as well as all of the ***predefined static or dynamic parameters***; it can be analyzed and dynamically modified or added to. By downloading the configuration file, for example, the autonomous agents are installed, via the interface IWMN (of the node NI) with the management node MN, in the nodes to be monitored from the management node MN, a specific command being used to install the autonomous agent SAA, as in the FIGURE, in the node NI. Each node to be monitored has its own files SL ("scanlog") of parameters, conditions and associated actions which allow it to control its own monitoring, while the management node also holds the status files of the nodes to be monitored as well as the parameter display files (a set of "trace" files TF). The updating of the list of the nodes in which an autonomous agent is installed is done automatically by the management node.

Boukobza at 4:59-5:9.

1652. Further examples include Boukobza at 5:40-46, 5:47-62.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1653. Boukobza discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

If oraDispatcherRejectedConnections increases greatly, monitor oraDispatcherState (BLOCKED, READY). If oraPrespawndSrvRejectedConnections increases greatly, monitor oraPrespawndState (BLOCKED, READY). If oraPrespawndSrvProcessorID is the process ID: make the connection with the consumption of system resources.

Boukobza at 33:1-7.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

1654. Boukobza discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730

Pat. at Cl. 19. For example:

Advantageously, for the application of the monitoring process according to the invention, ***the management node*** includes, among other things, a graphical user interface for the viewing of selected objects and the display of parameter value curves, ***a configuration file which contains all of the***

***configurations of the objects with the description of these objects*** as well as all of the predefined static or dynamic parameters, which file is analyzed and dynamically modified or added to, status files of the nodes to be monitored as well as parameter display files, the parameters measured being stored in a trace file so that they can be displayed by means of the graphical interface.

Boukobza at 3:60-4:4.

1655. Further examples include Boukobza at 21:8-14.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*database to store the policy for the agent.*

1656. Boukobza discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

Each specific module, SM1, SM2, ... , SMn, documents its predefined part by offering default lists for the selection of parameters to be measured (for example, parameter identifier, description of the measurement, period, ***storage or non-storage of the values in a "trace" file TF***, display or non-display of the values), of conditions to be evaluated on its parameters and of associated actions to be initiated. A parameter can also be a piece of information on the status of an object (OK, which in the preceding example corresponds to a green icon, warning or alert, which corresponds to an orange icon, and alarm or down, which corresponds to a red icon) or a measurement such as, for example, "the number of transactions per second". If a parameter condition is true, an action is initiated and the fact that an action has been initiated is noted in an action log file (action, level, date, parameter identifier, object identifier, node, etc.).

Boukobza at 5:47-62.

1657. Further examples include Boukobza at 3:60-4:4, 4:5-15, 4:59-63, 5:2-7.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1658. Boukobza discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

In this respect, the monitoring process mentioned in the preamble is noteworthy in that the monitoring is configured and then distributed in a filtered way from the management node to autonomous agents, an autonomous agent being installed in each node to be monitored in order, by providing intertype correlation, either to monitor as near as possible the different object types or all of the object of a domain called a global object, defined generically, or to feed back information to be displayed through ***the graphical interface*** of the management node, each agent comprising a plurality of specific modules specific to the different object types or to a particular domain, each specific module measuring static and dynamic parameters specific to the object type it monitors and collecting these measurements, testing conditions on these parameters relative to predefined thresholds and possibly initiating actions associated with these tested conditions, which parameters, conditions and actions are modifiable by the user of the management node.

Boukobza at 2: 21-38.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1659. Further examples include Boukobza at 4:39-42, 4:36-42, 4:42-59; Boukobza at Fig.

1.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

1660. Boukobza discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1661. Boukobza discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

Each specific module, SM1, SM2, ... , SMn, documents its predefined part by offering default lists for ***the selection of parameters to be measured*** (for example, parameter identifier, description of the measurement, period, storage or non-storage of the values in a "trace" file TF, display or non-display of the values), of conditions to be evaluated on its parameters and of associated actions to be initiated. A parameter can also be a piece of information on the status of an object (OK, which in the preceding example corresponds to a green icon, warning or alert, which corresponds to an orange icon, and alarm or down, which corresponds to a red icon) or a measurement such as, for example, "the number of transactions per second". If a parameter condition is true, an action is initiated and the fact that an action has been initiated is noted in an action log file (action, level, date, parameter identifier, object identifier, node, etc.).

Boukobza at 5:47-62.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

1662. Boukobza discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

In conclusion, according to the present monitoring process, the use of ***autonomous agents makes it possible to ensure the proper running of the applications monitored in all of the nodes by means of an autonomous and efficient decision processing*** applied locally to the objects to be processed, when necessary to very rapidly feed back the useful information from the nodes to be monitored to the management node, and to automatically initiate actions on certain conditions or possibly to recommend an action.

Boukobza at 34:52-60.

1663. Further examples include Boukobza at 3:30-39, 4:5-15, 6:30-35.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

1664. Boukobza discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

DisplayState (object\_ident, param\_ident, objtype, severity level, msg, line1val, line2val): to return the status of an object so that the interface GUI displays it. The interface GUI modifies the status of the icon associated with the object. "Objtype" is, for example, ORACLE, SYSTEM, TUXEDO, etc. "Msg" is detailed information on the status of the object: via the "button2" (middle), the interface GUI displays this information. ***The display action takes place on the management machine.*** "Line1val" and "line2val" (lines under the icon) are the values of the two lines: "NULL" means not to display the line under the icon.

Boukobza at 17:17-29.

1665. Further examples include Boukobza at 18:1-11, 34:43-51, 2:55-65, 5:63-6:14, 4:36-42, 3:30-39.

- b. *assigning a goal to a software agent; wherein the software agent*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*has its own runtime environment;*

1666. Boukobza discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

c. *is able to communicate with other software agents in the computer network*

1667. Boukobza discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

d. *is capable of perceiving its own state; and*

1668. Boukobza discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself;*

1669. Boukobza discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1670. Boukobza discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1671. Boukobza discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the computer network*

1672. Boukobza discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

1673. Boukobza discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1674. Boukobza discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

1675. Boukobza discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1676. Boukobza discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*component by the software agent in performing the predefined task; and*

1677. Boukobza discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

b. *constructing a topological representation of the computer network from the information.*

1678. Boukobza discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

1679. Boukobza discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1680. Boukobza discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

a. *The machine-readable storage medium of claim 33, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*determining uses a numerical method.*

1681. Boukobza discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1682. Boukobza discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1683. I expect to testify that Boukobza anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Boukobza discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-16.

**H. Anticipation by and/or Obviousness in View of United States Patent No. 6,665,262, filed 2/16/1999 and issued to Lindskog, et al. on 12/16/2003.**

1684. As explained in detail below and in the chart attached as Ex. A-18, Lindskog anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

1685. Lindskog discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example,

A system and ***method for distributing the fault management functions throughout a communications network***. Each node in the network includes an associated fault agent/ configuration agent pair. The fault agent receives alarm information and correlates the alarm information to ***identify a cause of a fault in the network***.

Lindskog at Abstract.

1686. Further examples include Lindskog at 3:11-14, 4:43-57, Fig. 1.

- b. *assigning a goal to a software [agent], wherein the software agent*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*has its own runtime environment*

1687. Lindskog discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example,

*Each FA 12 receives alarms, or events, from its associated network resources 18 (as indicated at 20) or from its subordinate FAs 12 (as indicated at 22). These alarms are used to identify faults that occur in the system 10, so that corrective actions can be taken. The receiving FA 12 then decides whether the alarm can be handled on the current level or if it must be forwarded (as indicated at 22) to the next higher agent level. Higher level FAs 12 (and CAs 14) are also able to draw conclusions based on information from only a subset of its subordinate FAs 12.*

Lindskog at 5:6-14.

1688. Further examples include Lindskog at 3:27-32, 8:59-65, 4:58-5:1, 3:36-50, 10:24-34, Fig.1, Fig. 4.

1689. To the extent this limitation is not disclosed by Lindskog, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.1.

c. *is able to communicate with other software agents in the computer network;*

1690. Lindskog discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example,

Once the alarm data reaches a fault agent at which the underlying fault can be handled, *the fault agent forwards the fault information to the associated configuration agent*. The configuration agent processes the fault information to generate reconfiguration data for correcting or otherwise reducing the effect of the fault and *sends the reconfiguration data to a network resource (ordering it to perform some action), or to at least one subordinate configuration agent*, depending on which nodes are affected by the proposed reconfiguration.

Lindskog at 3:36-44.

1691. Further examples include Lindskog at 5:66-6:11, 10:35-41, 11:56-67.

d. *is capable of perceiving its own state*

1692. Lindskog discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example,

Alarms that are to be handled at the current level are delivered to the associated CA 14 of the FA-CA pair 16 (as indicated at 24). *Each CA14 is assumed to be able to initiate the proper corrective measures or to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*implement appropriate system reconfigurations* in response to the received alarm. These measures and reconfigurations can be in response to a single received alarm or, using alarm correlation, to a specific sequence or group of received alarms. In addition, *the corrective measures and reconfigurations can include the performance of complicated calculations (e.g., involving various planning algorithms in knowledge-based systems)* that affect a number of underlying resources 18 or can be very simple, such as a decision not to act at all (i.e., a simple type of alarm filtering).

Lindskog at 5:42-55

1693. Further examples include Lindskog at 5:6-14, 11:56-62, 11:5-14, 3:36-44, 4:63-5:1.

e. *and is able to clone itself,*

1694. Lindskog discloses this element of claim 1, “and is able to clone itself.” ’730 Pat.

at Cl. 1. For example,

In addition, the use of autonomous and "intelligent" agents 12 and 14 allows the system to be scalable, wherein additional levels can be added without causing a corresponding increase in the risk of overloading the bandwidth of the fault management system, by avoiding the situation in which all fault data must be passed to a centralized control node.

Lindskog at 11:56-62.

1695. Further examples include Lindskog at 5:42-55.

1696. To the extent this limitation is not disclosed by Lindskog, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1697. Lindskog discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example,

Returning now to FIG. 2, when an update condition 108 or 114 is triggered in the event generator 44, the event generator 44 sends a message (as indicated at 46) to the event database 38, causing the event record for the generated event to be updated by changing the value of the value field and storing a new time in the time generated field. Whenever an event of an FA 12 is updated in the event database 38, the event dispatcher 40 is activated and the events are queued, as discussed above in connection with the subordinate FAs 12, one after another according to their priority and deadline. Thus, when Event n is generated in the FA 11, it is transmitted via the event dispatcher to the requesting agent CA 11, *in accordance with the original subscription request* (which was sent as indicated at 30). In this

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

manner, the appropriate CAs 14 in the system 10 are able to request and receive alarm information (events), in response to which ***the receiving CA14 is potentially able to take the necessary corrective actions.***

Lindskog '262 at 8:41-58.

1698. Further examples include Lindskog at 3:36-50, 10:24-34, 7:63- 8:16, Fig. 2, Fig. 3.

g. *monitoring the computer network;*

1699. Lindskog discloses this element of claim 1, “monitoring the computer network.”

'730 Pat. at Cl. 1. For example,

In accordance with the fault management architecture of the present invention, FA 12 in the RNM 62 sends individual subscription requests to the RNCs 64, to subscribe to certain alarm events in the respective RNCs 64. Each individual subscription request is sent once when it has been decided that a particular functionality should be activated. Then, potential problems in the network 60 associated with the particular functionality are constantly monitored until the subscription is canceled.

Lindskog at 10:63-11:4.

1700. Further examples include Lindskog at 11:47-56, 4:63-5:1, Fig. 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1701. Lindskog discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” '730 Pat. at Cl. 1. For example,

Once the alarm data reaches a fault agent at which the underlying fault can be handled, the fault agent forwards the fault information to the associated configuration agent. The configuration ***agent processes the fault information to generate reconfiguration data for correcting or otherwise reducing the effect of the fault and sends the reconfiguration data*** to a network resource (ordering it to perform some action), or to at least one subordinate configuration agent, depending on which nodes are affected by the proposed reconfiguration. Each subordinate configuration agent that receives the reconfiguration data generally performs further processing to generate more detailed reconfiguration data, which is then passed on to even lower level configuration agents or to underlying network resources. ***This process repeats until the reconfiguration is fully implemented.***

Lindskog at 3:36-50.

1702. Further examples include Lindskog at 3:64-4:10, 4:48-52.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1703. To the extent this limitation is not met, it would have been obvious in view of Goldman or Hellerstein. *See* Ex. A-3, limitation 1.7; Ex. A-20.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1704. Linskog discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example,

Alarms that are to be handled at the current level are delivered to the associated CA 14 of the FA-CA pair 16 (as indicated at 24). ***Each CA14 is assumed to be able to initiate the proper corrective measures or to implement appropriate system reconfigurations*** in response to the received alarm. These measures and reconfigurations can be in response to a single received alarm or, using alarm correlation, to a specific sequence or group of received alarms. In addition, the corrective measures and reconfigurations can include ***the performance of complicated calculations (e.g., involving various planning algorithms in knowledge-based systems)*** that affect a number of underlying resources 18 or can be very simple, such as a decision not to act at all (i.e., a simple type of alarm filtering).

Linskog at 5:42-55.

1705. Further examples include Linskog at 5:6-14, 8:7-16.

1706. To the extent this limitation is not met, it would have been obvious in view of Goldman or Hellerstein. *See* Ex. A-3, limitation 1.8; Ex. A-20.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1707. Linskog discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example,

Each FA 12 receives alarms, or events, from its associated network resources 18 (as indicated at 20) or from its subordinate FAs 12 (as indicated at 22). These alarms are used to identify faults that occur in the system 10, so that corrective actions can be taken. ***The receiving FA 12 then decides whether the alarm can be handled on the current level or if it must be forwarded (as indicated at 22) to the next higher agent level. Higher level FAs 12 (and CAs 14) are also able to draw conclusions*** based on information from only a subset of its subordinate FAs 12.

Linskog at 5:6-14.

1708. Further examples include Linskog at 5:42-55, Abstract.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1709. To the extent this limitation is not met, it would have been obvious in view of Goldman or Hellerstein. *See* Ex. A-3, limitation 1.9; Ex. A-20.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1710. Linskog discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example,

Whenever an event of a CA 14 is updated in the event database 38 (i.e., by storing a new cell plan), the event dispatcher 40 is activated and the events are queued one after another according to their priority and deadline. Thus, when Event 2 is generated in CA11, it is transmitted to the subordinate CA112 (as indicated at 48). In this manner, the CAs 14 in the system 10 are able to receive a high level cell plan, generate a new, more detailed cell plan for the subordinate CAs 14 and underlying resources 18, and transmit the new cell plan to the appropriate subordinate CAs 14 and resources 18. Similarly, with respect to received events that contain alarm data, ***the CAs 14 are able to evaluate the alarm, generate appropriate cell plans or other corrective measures, and transmit appropriate instructions to the subordinate CAs 14 and underlying resources 18 (as indicated at 48).***

Linskog at 10:19-34.

1711. Further examples include Linskog at 9:22-31, 5:42-55, 11:47-56.

1712. To the extent this limitation is not met, it would have been obvious in view of Goldman or Hellerstein. *See* Ex. A-3, limitation 1.10; Ex. A-20.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1713. Linskog discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example,

In addition, ***the use of autonomous and "intelligent" agents 12 and 14*** allows the system to be scalable, wherein additional levels can be added without causing a corresponding increase in the risk of overloading the bandwidth of the fault management system, by avoiding the situation in which all fault data must be passed to a centralized control node.

Linskog at 11:56-62.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1714. Further examples include Linskog at 11:5-24, Claim 4, 12:44-47, Claim 5, 12:48-51, 5:6-14, 5:55-65, Abstract.

1715. Additionally, this limitation would have been obvious in combination with Turek. *See* Ex. A-1 at limitation 1.11.

**2. Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1716. Linskog discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example,

Once the appropriate corrective measures and system reconfigurations are determined by the CA 14, ***corresponding instructions are sent*** from the CA 14 to the underlying network resources 18 (as indicated at 26) and/or to the subordinate CAs 14 (as indicated at 28), depending on where corrective actions are needed. Upon receipt of ***such instructions*** at a subordinate CA 14, additional processing can be performed, with any additional instructions being sent to the underlying resources 18 (as indicated at 26) and/or to even more subordinate CAs 14 (as indicated at 28).

Linskog at 5:55-65.

1717. Further examples include Linskog at Claim 10, 13:1-3, 10:24-34, 6:66-7:6, Fig. 2.

1718. To the extent this limitation is not disclosed by Linskog ’262, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 2.0.

**3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1719. Linskog discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example,

Each FA 12 receives alarms, or events, from its associated network resources 18 (as indicated at 20) or from its subordinate FAs 12 (as indicated at 22). These alarms are used to identify faults that occur in the system 10, so that corrective actions can be taken. The receiving FA 12 then decides whether the alarm can be handled on the current level or if it must be forwarded (as indicated at 22) to the next higher agent level. ***Higher level***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*FAs 12 (and CAs 14) are also able to draw conclusions based on information from only a subset of its subordinate FAs 12.*

Lindskog at 5:6-14.

1720. Further examples include Lindskog at 6:15-23, 3:36-50.

- b. *constructing a topological representation of the computer network from the information.*

1721. Lindskog discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example,

Similarly, the configuration agents include an event receiver for receiving both fault information from associated fault agents and high level reconfiguration data from higher level configuration agents. The configuration agents further include an event generator for generating the reconfiguration data necessary to reduce the effect of a detected fault and for generating more detailed reconfiguration data from the high level data that is received from supervising configuration agents. Once the reconfiguration data is generated, the event generator updates the configuration information that is stored in an event database, and, as a result, ***the event dispatcher sends the updated configuration information to the subordinate configuration agents and/or to underlying network resources for implementation.***

Lindskog at 3:64-4:10.

1722. Further examples include Lindskog at 5:42-54.

1723. To the extent this limitation is not disclosed by Lindskog ’262, it would have been obvious to combine Lindskog with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

**4. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1724. Lindskog discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example,

These measures and reconfigurations can be in response to a single received alarm or, using alarm correlation, to a specific sequence or group of received alarms. In addition, ***the corrective measures and reconfigurations can include the performance of complicated calculations*** (e.g., involving various planning algorithms in knowledge-based systems) that affect a number of underlying resources 18 or can be very simple, such as a decision not to act at all (i.e., a simple type of alarm filtering).

Lindskog at 5:47-54.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1725. Further examples include Lindskog at 7:40-53.

1726. To the extent this limitation is not disclosed by Lindskog '262, it would have been obvious to combine Lindskog '262 with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

1727. Furthermore, modeling using numerical models such as the Dijkstra self-stabilization algorithm were well known at the time of invention. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek.

5. ***Claim 6***

a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1728. Lindskog discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example,

Each subordinate configuration agent that receives the reconfiguration data generally performs further processing to generate more detailed reconfiguration data, which is then passed on to even lower level configuration agents or to underlying network resources. ***This process repeats until the reconfiguration is fully implemented.***

Lindskog at 3:44-50.

1729. Further examples include Lindskog at Abstract, 5:47-55.

1730. To the extent this limitation is not disclosed by Lindskog '262, it would have been obvious to combine Lindskog '262 with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2).

1731. Furthermore, modeling using numerical models such as the Dijkstra self-stabilization algorithm were well known at the time of invention. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek.

6. ***Claim 7***

a. *A computer network, comprising:*

1732. Lindskog discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example,



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In particular, the distributed fault management architecture the present invention decentralizes the fault management operations and increases the robustness of the fault management system, while permitting faults to be addressed in real time. Essentially, fault management capabilities are distributed throughout the various levels of a communications network. Maximum efficiency is then achieved by performing fault processing, and making necessary corrections, at the lowest possible level. In addition, the decentralization of the fault management operations allows scalability (i.e., the ability to continue operations as new equipment or levels are added to the network) of the system to a much larger degree than with a centralized fault management solution.

Lindskog at 4:43-57.

1733. Further examples include Lindskog at 4:58-5:1, 6:33-40, Fig. 1, Fig. 2.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1734. Lindskog discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

1735. Lindskog discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

1736. Lindskog discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

1737. Lindskog discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

- f. *is able to clone itself;*

1738. Lindskog discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

- g. *an agent support mechanism embodied in hardware to provide*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

1739. Linskog discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example,

These FA-CA pairs ***16 can be device***, such as a radio network associated with a particular controller or radio network manager in a cellular system, or with a logical entity, such as a cell-pair or a location area.

Linskog at 4:63-66.

1740. Further examples include Linskog at 3:11-23.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1741. Linskog discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

1742. Linskog discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

1743. Linskog discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1744. Linskog discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1745. Lindskog discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1746. Lindskog discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1747. Lindskog discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example,

Instead, *each FA 12 (or CA 14) first sends a subscription request to a subordinate FA 12* (or associated FA 12) individually subscribing to specific events (i.e., alarms) so as to obtain maximum flexibility. Using this subscription process, the FA 12 informs the subordinate FA 12 which alarms are of interest to the operating network at a particular time (e.g., for this hour). Thus, *the subordinate FA 12 only forwards alarm information to agents that have requested such information through a subscription request*. This function avoids the consumption of bandwidth for sending data that are not relevant for the moment. This can also be thought of as an alarm filtering operation.

Lindskog at 6:3-14.

1748. Further examples include Lindskog at 6:15-33, 4:58- 5:5, 11:56-67.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism comprises a secure communications protocol.*

1749. Linskog discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example,

Referring now to FIG. 1, there is illustrated a simplified diagram of the distributed fault management system 10 of the present invention. The system 10 includes any number of fault agents (FA) 12 and configuration agents (CA) 14 operating on different levels. Usually, ***the FAs 12 and CAs 14 are arranged in pairs***. These FA-CA pairs 16 can be associated with a particular device, such as a radio network controller or radio network manager in a cellular system, or with a logical entity, such as a cell-pair or a location area. ***Each FA-CA pair 16 supervises one or more underlying network resources 18 and/or subordinate FA-CA pairs 16.***

Linskog at 4:58-5:1.

1750. Further examples include Linskog at 6:3-14.

1751. To the extent this limitation is not disclosed by Linskog, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 12.0.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

1752. Linskog discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example,

Once the alarm data reaches a fault agent at which the underlying fault can be handled, the fault agent forwards the fault information to the associated configuration agent. The configuration ***agent processes the fault information to generate reconfiguration data*** for correcting or otherwise reducing the effect of the fault and sends the reconfiguration data to a network resource (ordering it to perform some action), or to at least one subordinate configuration agent, depending on which nodes are affected by the proposed reconfiguration. Each subordinate configuration ***agent that receives the reconfiguration data generally performs further processing to generate more detailed reconfiguration data***, which is then passed on to even lower level configuration agents or to underlying network resources. This process repeats until the reconfiguration is fully implemented.

Linskog at 3:36-50.

1753. Further examples include Linskog at 10:36-47,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1754. Linskog discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example,

When an event is received by CA11, the event is immediately passed on to a processing unit 36 in CA11. ***The processing unit 36 checks to ensure that the event is valid*** (e.g., by accessing the events database 38), and if so, the event is forwarded to the event generator 44. The event generator 44 is responsible for carrying out all computations needed to submit proper configuration changes to the underlying CAs 14 and network resources 18. These computations can be relatively complex, especially for the generation or extraction of a new cell plan.

Linskog at 9:22-31.

1755. Further examples include Linskog at 6:33-44, 7:7-26.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1756. Linskog discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example,

First an agent 12 or 14, in this case CA11, that desires event information from network resources 18 or other agents 12 or 14 (CA11's associated fault agent FA11 in this case) subscribes to event data from FA11 by sending one or more subscription requests to FA11 (as indicated at 30). The subscription request, as shown in its inset view, contains: (1) a pointer to an event receiver 32 of the requesting agent, here CA11 (used as a subscriber address); (2) an identification of the event for which information is requested (Event n in this example); (3) the priority of receiving the event information (e.g., 1 to 10, with 1 being the highest priority); and (4) a deadline indicating the latest time at which the event information should be passed. After a subscription request is made, the requesting agent can suspend the request, resume the request (once suspended), or cancel the request.

Linskog at 6:17-32.

1757. Further examples include Linskog at 6:41-65, Fig. 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1758. To the extent this limitation is not disclosed by Lindskog, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 18.0.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

1759. Lindskog discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example,

The processing unit 36 interacts with an events database 38 to determine what actions need to be taken to properly respond to the request. ***The events database 38 contains event records for all of the possible events that may occur at agent FA11.*** Each event record, as shown in the inset view of Event n, contains: (1) a value for that event (e.g., providing refined event information, indicating the number of times the event has occurred, and so on) (2) an indication of the time at which the event was last generated (i.e., occurred); (3) ***a list of current subscribers*** (ordered according to the subscribers' importance, which is a function of the above-mentioned priority and deadline information); and (4) a listing of the required input events (i.e., from subordinate FAs 12) to be able to generated the event in question.

Lindskog at 6:41-54.

1760. Further examples include Lindskog at 11:47-67.

1761. To the extent this limitation is not disclosed by Lindskog, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 19.0.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

1762. Lindskog discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example,

Referring now to FIG. 2, there is shown a diagram illustrating the preferred operation of an FA12 in accordance with the present invention. First an agent 12 or 14, in this case CA11, that desires event information from network resources 18 or other agents 12 or 14 (CA11's associated fault agent FA11 in this case) subscribes to event data from FA11 by sending one or more subscription requests to FA11 (as indicated at 30). ***The subscription request***, as shown in its inset view, contains: (1) a pointer to an event receiver 32 of the requesting agent, here CA11 (used as a subscriber

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

address); (2) an identification of the event for which information is requested (Event n in this example); (3) the priority of receiving the event information (e.g., 1 to 10, with 1 being the highest priority); and (4) a deadline indicating the latest time at which the event information should be passed. After a subscription request is made, the requesting agent can suspend the request, resume the request (once suspended), or cancel the request.

Lindskog at 6:15-32.

1763. Further examples include Lindskog at 9:14-21, 9:51-67, Fig. 2, Fig. 4.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1764. Lindskog discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example,

***This event generator process can be performed in many ways.*** For example, in EHPT's centralized Fault Management expert (FMX) system, a graphical rule-based if-then language is used to perform event filtering and correlation. In the preferred embodiment, the event generator 44 employs a technique that uses so-called function charts, typically referred to as the GRAFCET formalism. GRAFCET is an IEC standard for sequential control, it supports high-level hierarchical control design, and ***by its graphical appearance, it is easy to configure and understand.*** In addition, because GRAFCET is particularly well suited for control applications, it can be used both for event filtering and correlation (in FAs) and for configuration control (in CAs).

Lindskog at 7:49-62.

1765. Further examples include Lindskog at 6:41-57.

1766. To the extent this limitation is not met, it would have been obvious in view of Turek. See Ex. A-1 at limitation 22.0.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

1767. Lindskog discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1768. Linskog discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example,

As a result of the transition at step 204, an ordinary step 206 is activated, whereupon the event generator 44 develops a new plan for the subordinate CA111 that is in accordance with the higher level (less detailed) cell plan contained in Event 3 from the CA1. If a "plan ready for CA111" transition is triggered at step 208, then an "update" step 210 is activated, whereupon ***Event 1 is updated in the event database 38 with a new value, which contains the new cell plan for CA11.*** In the parallel branch, an ordinary step 214 is activated simultaneously with the step 206, whereupon the event generator 44 develops a new plan for the subordinate CA112 in accordance with the higher level (less detailed) cell plan contained in Event 3 from the CA1. If a "plan ready for CA112" transition is triggered at step 216, then an update step 218 is activated, whereupon Event 2 is updated in the event database 38 with a new value, which contains the new cell plan for the CA112.

Linskog at 9:51-67.

1769. Further examples include Linskog at 3:36-50, 7:28-39, 6:41-57.

1770. To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 26.0.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

1771. Linskog discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example,

In addition, ***the use of autonomous and "intelligent" agents 12 and 14*** allows the system to be scalable, wherein additional levels can be added without causing a corresponding increase in the risk of overloading the bandwidth of the fault management system, by avoiding the situation in which all fault data must be passed to a centralized control node. Moreover, the distributed intelligence of the system 60 enables robust fault management handling. ***Flexibility can also be achieved by allowing the***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*agents 12 and 14 to be interconnected in almost any way* by using, for example, standard communication mechanisms such as CORBA.

Lindskog at 11:56-67.

1772. Further examples include Lindskog at 5:6-14, 10:24-34, 6:41-57.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

1773. Lindskog discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example,

When an event is received by CA11, the event is immediately passed on to a processing unit 36 in CA11. ***The processing unit 36 checks to ensure that the event is valid*** (e.g., by accessing the events database 38), and if so, the event is forwarded to the event generator 44.

Lindskog at 9:22-26.

1774. Further examples include Lindskog at 7:7-27, 7:40-49, 9:22-26, 3:11-23, 4:63-66, Fig. 2.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1775. Lindskog discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

1776. Lindskog discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

1777. Lindskog discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is able to clone itself;*

1778. Lindskog discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1779. Lindskog discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1780. Lindskog discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1781. Lindskog discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a predictive algorithm;*

1782. Lindskog discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1783. Lindskog discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network by replacing the assigned goal based on the optimal policy.*

1784. Lindskog discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

**20. Claim 31**

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1785. Lindskog discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

**21. Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1786. Lindskog discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

1787. Lindskog discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task;*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*and*

1788. Lindskog discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1789. Lindskog discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1790. Lindskog discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1791. Lindskog discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1792. I expect to testify that Lindskog anticipates and/or renders obvious each Asserted Claim of the '730 Patent. A claim chart illustrating that Lindskog discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-18.

**I. Anticipation by and/or Obviousness in View of United States Patent No.**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**6,675,128, filed 9/30/1999 and issued to Hellerstein, et al. on 1/6/2004.**

1793. As explained in detail below and in the chart attached as Ex. A-20, Hellerstein anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

1794. Hellerstein discloses the preamble of claim 1, "[a] method of managing a computer network, comprising." '730 Pat. at Cl. 1. For example:

***The present invention provides methods and apparatus that reduce the burden on administrators for performance management.*** The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.

Hellerstein at 2:57-64.

1795. Further examples include Hellerstein at 6:28-36, 9:23-38.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1796. Hellerstein discloses this element of claim 1, "assigning a goal to a software [agent], wherein the software agent has its own runtime environment." '730 Pat. at Cl. 1. For example:

As will be illustrated in detail below, a performance management system of the invention may comprise a system manager to support the policies described herein which is responsible for one or more model-based policy agents that respectively reside on one or more managed elements. The system manager provides an interface for the administrator through which policies are authored and messages are reported. The model-based policy agent may preferably have the following components: an agent side policy controller, a model constructor, a threshold constructor, a control policy evaluator, and an action executor. ***In such an embodiment, the policy controller provides overall control of the agent. The model constructor estimates the unknown constants in the metric models that provide a way to predict future values and a method of determining the distribution of metric values for a particular system, e.g., time-of-day, etc.*** The threshold constructor uses the metric models and the false alarm policies to determine the value of alarm and warning thresholds. The control policy evaluator provides a mechanism for determining if the LHS of a policy is satisfied at a specific time. The action executor provides a means for executing the right-hand side of policies.

Hellerstein at 4:52-5:6.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1797. Further examples include Hellerstein at 7:51-59; Hellerstein at Fig. 1.

1798. As an additional example, the manger is also an agent and has goals assigned to it:

On each managed element is also a model-based policy agent 120 that enforces performance management policies, as will be explained in detail below. The managed elements communicate with *a manager 140 that is controlled by an administrative user 160* who is responsible for authoring and ensuring the distribution of management policies through the policy authoring, distribution, and reporting component 145. *In particular, the manager 140 provides an administrator 160 a place to enter information into the following repositories: model reconstruction policies 205, alarm policies 225, control policies 210, and action trigger policies 220.*

Hellerstein at 6:37-48.

1799. Further examples include Hellerstein at Fig. 3.

c. *is able to communicate with other software agents in the computer network;*

1800. Hellerstein discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

Also, it is to be appreciated that *the computer systems of the managed elements and the manager may be in communication with each other via conventional communication links*, e.g., local area network, wide area network, etc. Further, as mentioned, more than one computer system may be employed to implement the components illustrated in a managed element or a manager.

Hellerstein at 8:37-42.

1801. Further examples include Hellerstein at 6:37-44; Hellerstein at Fig. 1.

1802. To the extent that this limitation is not expressly disclosed by Hellestein, this limitation would have been obvious to a person of ordinary skill in the art because the coordination of multiple agents implies an ability to communicate with other agents.

d. *is capable of perceiving its own state*

1803. Hellerstein discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

As will be illustrated in detail below, a performance management system of the invention may comprise a system manager to support the policies described herein which is responsible for one or more model-based policy agents that respectively reside on one or more managed elements. The system manager provides an interface for the administrator through which

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

policies are authored and messages are reported. The model-based policy agent may preferably have the following components: an agent side policy controller, a model constructor, a threshold constructor, a control policy evaluator, and an action executor. *In such an embodiment, the policy controller provides overall control of the agent. The model constructor estimates the unknown constants in the metric models that provide a way to predict future values and a method of determining the distribution of metric values for a particular system, e.g., time-of-day, etc.* The threshold constructor uses the metric models and the false alarm policies to determine the value of alarm and warning thresholds. The control policy evaluator provides a mechanism for determining if the LHS of a policy is satisfied at a specific time. The action executor provides a means for executing the right-hand side of policies.

Hellerstein at 4:52-5:6.

1804. Further examples include Hellerstein at 6:17-26; Hellerstein at Fig. 1.

1805. As an additional example, the manager is also an agent, and similarly aware of its own state:

*Platform services 110 similar to the services provided for in the managed elements 100 may be provided in the manager 140.* However, other services specific to the manager may be provided. One or more of these platform services allow the manager to communicate with the managed elements.

Hellerstein at 6:53-58.

e. *and is able to clone itself,*

1806. Hellerstein discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

In particular, the manager 140 provides an administrator 160 a place to enter information into the following repositories: model reconstruction policies 205, alarm policies 225, control policies 210, and action trigger policies 220. *Policy authoring, distribution, and reporting is well understood in the art.* It involves, for example: entering information in a graphical user interface to be Stored in a database, *distributing databases to other machines, and reporting the results of events that take place on remote machines.* Platform services 110 similar to the services provided for in the managed elements 100 may be provided in the manager 140. However, other services specific to the manager may be provided. One or more of these platform services allow the manager to communicate with the managed elements.

Hellerstein at 6:44-57.

1807. Further examples include Hellerstein at Fig. 1.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1808. As an additional example, the agents also clone themselves, because they send copies of their event logs to the manager:

Referring now to FIG. 2, a block diagram illustrating a model-based policy agent 120 according to an exemplary embodiment of the present invention is shown. In the model-based policy agent, an agent side policy controller 200 provides overall control and interacts with the platform services to ***update local copies of policy repositories and to report alarms and warnings to the manager 140.***

Hellerstein at 7:10-16.

1809. Further examples include Hellerstein at Fig. 2.

1810. To the extent that this limitation is not expressly disclosed by Hellerstein, this limitation would have been obvious to a person of ordinary skill in the art because when agents are coordinating to accomplish large tasks (such as modelling predicted network behavior) together, it would be beneficial for the agents to clone themselves.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1811. Hellerstein discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

As will be illustrated in detail below, a performance management system of the invention may comprise a system manager to support the policies described herein which is responsible for one or more model-based policy agents that respectively reside on one or more managed elements. The system manager provides an interface for the administrator through which policies are authored and messages are reported. The model-based policy agent may preferably have the following components: an agent side policy controller, a model constructor, a threshold constructor, a control policy evaluator, and an action executor. ***In such an embodiment, the policy controller provides overall control of the agent. The model constructor estimates the unknown constants in the metric models that provide a way to predict future values and a method of determining the distribution of metric values for a particular system, e.g., time-of-day, etc.*** The threshold constructor uses the metric models and the false alarm policies to determine the value of alarm and warning thresholds. The control policy evaluator provides a mechanism for determining if the LHS of a policy is satisfied at a specific time. The action executor provides a means for executing the right-hand side of policies.

Hellerstein at 4:52-5:6.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1812. Further examples include Hellerstein at Fig. 1.

1813. As an additional example, the manager is also an agent and its goals are a programmatic expression of a predefined task:

On each managed element is also a model-based policy agent 120 that enforces performance management policies, as will be explained in detail below. The managed elements communicate with *a manager 140 that is controlled by an administrative user 160 who is responsible for authoring and ensuring the distribution of management policies through the policy authoring, distribution, and reporting component 145*. In particular, the manager 140 provides an administrator 160 a place to enter information into the following repositories: model reconstruction policies 205, alarm policies 225, control policies 210, and action trigger policies 220.

Hellerstein at 6:37-48.

g. *monitoring the computer network;*

1814. Hellerstein discloses this element of claim 1, “monitoring the computer network.”

’730 Pat. at Cl. 1. For example:

Generally, existing practice for detecting performance and availability problems consists of the following steps: (i) determining a set of metrics to monitor that indicate the presence of problems (e.g., CPU utilization, error rates, transaction request rates); (ii) establishing thresholds on the values of these metrics based on past experience; (iii) using management system software to detect threshold violations; and (iv) responding to threshold violations by taking actions (e.g., adjusting user priorities, restricting the admission of traffic into the network).

Hellerstein at 1:20-29.

1815. Further examples include Hellerstein at 4:52-5:6.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1816. Hellerstein discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. The methods and apparatus use models of metric values to construct and enforce: (1) *alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload*; and (2) warning policies based on the *probability of violating an alarm policy within a time horizon*.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*It is to be appreciated that a performance management system of the present invention preferably utilizes forecasting models (e.g., analysis of variance and time series models) to capture non-stationarities (e.g., time-of-day variations) and time-serial dependencies.*

Hellerstein at 2:57-3:2.

1817. Further examples include Hellerstein at 3:32-43, 4:52-5:18.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1818. Hellerstein discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the *probability of violating an alarm policy within a time horizon*.

*It is to be appreciated that a performance management system of the present invention preferably utilizes forecasting models (e.g., analysis of variance and time series models) to capture non-stationarities (e.g., time-of-day variations) and time-serial dependencies.*

Hellerstein at 2:57-3:2.

1819. Further examples include Hellerstein at 3:32-43, 4:52-5:18.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1820. Hellerstein discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.

*It is to be appreciated that a performance management system of the present invention preferably utilizes forecasting models (e.g., analysis of variance and time series models) to capture non-stationarities (e.g., time-of-day variations) and time-serial dependencies.*

Hellerstein at 2:57-3:2.

1821. Further examples include Hellerstein at 3:32-43, Hellerstein at 4:52-5:18.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1822. Hellerstein discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.

***It is to be appreciated that a performance management system of the present invention preferably utilizes forecasting models (e.g., analysis of variance and time series models) to capture non-stationarities (e.g., time-of-day variations) and time-serial dependencies.***

Hellerstein at 2:57-3:2.

1823. Further examples include Hellerstein at 3:32-43, 4:52-5:18.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1824. Hellerstein discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example, if the agent cannot perform the predefined task, it can request further policy from itself by modelling a new policy:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.

***It is to be appreciated that a performance management system of the present invention preferably utilizes forecasting models (e.g., analysis of variance and time series models) to capture non-stationarities (e.g., time-of-day variations) and time-serial dependencies.***

Hellerstein at 2:57-3:2.

1825. Further examples include Hellerstein at 3:32-43, 4:52-5:18, 7:10-16; Hellerstein at

Fig. 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1826. To the extent this limitation is not explicitly disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art. Hellerstein contemplates software agents relying policies from other agents, and discloses agents exchanging messages between one another. It would have been obvious to a person of ordinary skill in the art at the time of invention to modify Hellerstein so that an agent may request a policy from another entity (other agents, the manager, etc.) when it requires additional functionality, or when another agent reports superior results.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1827. Hellerstein discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

***The present invention preferably deals with two kinds of policies.*** The first kind of policies, referred to as "control policies," address the essence of the left-hand side of existing policies: identifying the metrics to be tested and the directional change in these metrics that constitutes an exceptional situation. The second kind of policies, referred to as "meta policies," determine how control policies are interpreted. Meta policies address the following: (a) the acceptable level of false alarms; (b) the choice of alarm threshold (based on metric models and false alarm policies); (c) when models should be reconstructed; and (d) when warnings are generated.

Hellerstein at 3:32-43.

1828. Further examples include Hellerstein at 4:52-5:6.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1829. Hellerstein discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

Generally, existing practice for detecting performance and availability problems consists of the following steps: (i) determining a set of metrics to monitor that indicate the presence of problems (e.g., CPU utilization, error rates, transaction request rates); (ii) establishing thresholds on the values of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

these metrics based on past experience; (iii) using management system software to detect threshold violations; and (iv) responding to threshold violations by taking actions (e.g., adjusting user priorities, restricting the admission of traffic into the network).

Hellerstein at 1:20-29.

1830. Further examples include Hellerstein at 4:52-5:6.

- b. *constructing a topological representation of the computer network from the information.*

1831. Hellerstein discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, *in configuration, topology, and workload*; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.

Hellerstein at 2:57-64.

1832. To the extent this limitation is not disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art at the time that when a system is aware of when there is a change in the topology of the network, it is because the system has constructed a topological representation of the network.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1833. Hellerstein discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.

*It is to be appreciated that a performance management system of the present invention preferably utilizes forecasting models (e.g., analysis of variance and time series models) to capture non-stationarities (e.g., time-of-day variations) and time-serial dependencies.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Hellerstein at 2:57-3:2.

1834. Further examples include Hellerstein at 3:32-43, 4:52-5:18.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1835. Hellerstein discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6.

6. ***Claim 7***

- a. *A computer network, comprising:*

1836. Hellerstein discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

***The present invention provides methods and apparatus that reduce the burden on administrators for performance management.*** The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.

Hellerstein at 2:57-64.

1837. Further examples include Hellerstein at 6:28-36, 9:23-38.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1838. Hellerstein discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

1839. Hellerstein discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

1840. Hellerstein discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

1841. Hellerstein discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

1842. Hellerstein discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1843. Hellerstein discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

Referring now to FIG. 1, a block diagram illustrating a performance management System according to an exemplary embodiment of the present invention is shown. In the system as shown, one or more business users 150 (150-1 through 150-N) respectively interact with one or more managed elements 100 (100-1 through 100-N). The number of business users may also differ from the number of managed elements. ***Each managed element may have one or more business applications 130 (130-1 through 130-K) that utilize platform services 110.***

Hellerstein at 6:17-26.

1844. Further examples include Hellerstein at 7:51-59; Hellerstein at Figs. 1, 3.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1845. Hellerstein discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*of a network component*

1846. Hellerstein discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the prediction;*

1847. Hellerstein discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1848. Hellerstein discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1849. Hellerstein discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1850. Hellerstein discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

1851. Hellerstein discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

Also, it is to be appreciated that ***the computer systems of the managed elements and the manager may be in communication with each other via conventional communication links***, e.g., local area network, wide area network, etc. Further, as mentioned, more than one computer system may be employed to implement the components illustrated in a managed element or a manager.

Hellerstein at 8:37-42.

1852. Further examples include Hellerstein at 6:37-44; Hellerstein at Fig. 1.

1853. To the extent that this limitation is not expressly disclosed by Hellerstein, this limitation would have been obvious to a person of ordinary skill in the art because the coordination of multiple agents implies an ability to communicate with other agents.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1854. Hellerstein discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

1855. Hellerstein discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

Referring now to FIG. 1, a block diagram illustrating a performance management System according to an exemplary embodiment of the present invention is shown. In the system as shown, one or more business users 150 (150-1 through 150-N) respectively interact with one or more managed

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

elements 100 (100-1 through 100-N). The number of business users may also differ from the number of managed elements. ***Each managed element may have one or more business applications 130 (130-1 through 130-K) that utilize platform services 110.***

Hellerstein at 6:17-26.

1856. Further examples include Hellerstein at 7:51-59; Hellerstein at Figs. 1, 3.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1857. Hellerstein discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

Referring now to FIG. 1, a block diagram illustrating a performance management System according to an exemplary embodiment of the present invention is shown. In the system as shown, one or more business users 150 (150-1 through 150-N) respectively interact with one or more managed elements 100 (100-1 through 100-N). The number of business users may also differ from the number of managed elements. ***Each managed element may have one or more business applications 130 (130-1 through 130-K) that utilize platform services 110.***

Hellerstein at 6:17-26.

1858. Further examples include Hellerstein at 6:37-58; Hellerstein at Fig. 1.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1859. Hellerstein discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

We use the term "alarm threshold" for the metric value that, if exceeded, results in either the ***generation of an alarm*** or a management action (***e.g., terminate a process***). Existing approaches check for threshold violations and, when these violations occur, initiate the action specified in the right-hand side of the policy.

Hellerstein at 1:47-53.

1860. Further examples include Hellerstein at 9:9-21.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

1861. Hellerstein discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example, each agent runtime environment must necessarily store a list of running processes; thus Hellerstein satisfies this limitation under NetFuel’s application of the claims.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

1862. Hellerstein discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

Policy authoring, distribution, and reporting is well understood in the art. It involves, for example: ***entering information in a graphical user interface to be stored in a database, distributing databases to other machines,*** and reporting the results of events that take place on remote machines.

Hellerstein at 6:48-53.

1863. Further examples include Hellerstein at 7:47-51.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1864. Hellerstein discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

Existing art provides for policy authoring and execution. That is, ***administrators typically have a graphical user interface through which they specify policy metrics, threshold values, relational operators, and actions.*** The management system acquires the data necessary to test the left-hand side of a policy and to execute the right-hand side of a policy.

Hellerstein at 1:66-2:4.

1865. Further examples include Hellerstein at 6:48-53.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

16. **Claim 24**

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

1866. Hellerstein discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. **Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1867. Hellerstein discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. ***The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.***

Hellerstein at 2:57-64.

1868. Further examples include Hellerstein at 5:45-50.

18. **Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

1869. Hellerstein discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

The present invention provides methods and apparatus that reduce the burden on administrators for performance management. ***The methods and apparatus use models of metric values to construct and enforce: (1) alarm policies that adjust automatically to changes, for example, in configuration, topology, and workload; and (2) warning policies based on the probability of violating an alarm policy within a time horizon.***

Hellerstein at 2:57-64.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1870. Further examples include Hellerstein at 5:45-50.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

1871. Hellerstein discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

Referring now to FIG. 1, a block diagram illustrating a performance management System according to an exemplary embodiment of the present invention is shown. In the system as shown, one or more business users 150 (150-1 through 150-N) respectively interact with one or more managed elements 100 (100-1 through 100-N). The number of business users may also differ from the number of managed elements. ***Each managed element may have one or more business applications 130 (130-1 through 130-K) that utilize platform services 110.***

Hellerstein at 6:17-26.

1872. Further examples include Hellerstein at 7:51-59; Hellerstein at Figs. 1, 3.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1873. Hellerstein discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

1874. Hellerstein discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

1875. Hellerstein discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is able to clone itself;*

1876. Hellerstein discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1877. Hellerstein discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1878. Hellerstein discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1879. Hellerstein discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a predictive algorithm;*

1880. Hellerstein discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1881. Hellerstein discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network by replacing the assigned goal based on the optimal policy.*

1882. Hellerstein discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. **Claim 31**

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1883. Hellerstein discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. **Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1884. Hellerstein discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

1885. Hellerstein discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. **Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task;*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*and*

1886. Hellerstein discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1887. Hellerstein discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1888. Hellerstein discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1889. Hellerstein discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1890. I expect to testify that Hellerstein anticipates and/or renders obvious each Asserted Claim of the '730 Patent. A claim chart illustrating that Hellerstein discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-20.

**J. Anticipation by and/or Obviousness in View of *Distributed Management with Mobile Components*, by Kasteleijn, et al., published in the May 1999 Proceedings of the Sixth IFIP/IEEE International Symposium on Integrated**



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Network Management.**

1891. As explained in detail below and in the chart attached as Ex. A-2, Kasteleijn anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

1892. Kasteleijn discloses the preamble of claim 1, "[a] method of managing a computer network, comprising." '730 Pat. at Cl. 1. For example:

The distributed management framework (DMF) presented in this paper provides an environment which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates. We present an object-oriented, Java-based implementation of the DMF and describe applications developed on this platform.

Kasteleijn at 857.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1893. Kasteleijn discloses this element of claim 1, "assigning a goal to a software [agent], wherein the software agent has its own runtime environment." '730 Pat. at Cl. 1. For example:

The distributed management architecture presented in this paper provides an ***environment*** which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, ***management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates.*** Management tasks are supported by a distributed directory and secure communications. The architecture does not force the programmer to design its management applications according to any given programming paradigm. Rather, the programmer is free to use paradigms such as client-server and cooperating peers. Mixed solutions are also possible.

Kasteleijn at 858.

1894. Further examples include Kasteleijn at 858, 859–860, 861–862, 864, 866; Kasteleijn at Figs. 1, 2, 3.

c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network;*

1895. Kasteleijn discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

## **2. Distributed Management Framework**

Distributed Management Framework (DMF) provides an object-oriented architecture where lightweight, mobile management applications can be dynamically deployed for distributed management. As shown in Figure 2, the Distributed Management Framework consists of a collection of *distributed management nodes* (DMNs), where each DMN provides the execution environment and supporting services for *management components* (MCs), i.e., the management applications. ***The DMNs form a network of peers over which hierarchies or cooperating sets of management components can be run. A distributed directory, maintained by each DMN is used in locating management components and services.*** Components and configuration of the DMF are stored in a set of replicated *code and configuration stores* (CCSs), from where they can be downloaded to the DMNs as required. In the following, we describe these components in detail.

Kasteleijn at 4.

1896. Further examples include Kasteleijn at 6; Kasteleijn at Fig. 3.

d. *is capable of perceiving its own state*

1897. Kasteleijn discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

### **Core Services**

The core services component provides the essential services for all components of the DMN.

The transport service is responsible for moving MCs between DMNs. **When MCs are moved, they can keep their internal state including the results of the completed management operations.** The communication service provides seamless remote and local communication between the management components. Communication includes not only messaging but also event forwarding, remote method invocation and exception forwarding.

Kasteleijn at 862.

e. *and is able to clone itself,*

1898. Kasteleijn discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 1 shows an enterprise network which is partitioned into several sub-networks (subnets). We assume that all relevant devices in the system support our architecture. The goal of the example management application is to determine the status of all available NFS daemons running in the enterprise network. The application, running possibly as a background troubleshooter, does not have a priori knowledge of the topology of the network or the location of the NFS servers.

Kasteleijn at 858.

1899. Further examples include Kasteleijn at 859-860; Kasteleijn at Fig. 1.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1900. Kasteleijn discloses this element of claim 1, "and wherein the goal is a programmatic expression of a predefined task for the software agent." '730 Pat. at Cl. 1. For example:

In our scenario shown in Figure 4, the EnterpriseManager listens for "delay threshold exceeded at LAN NYC" events from other MCs in the DMF. When it receives such an event, the following set of actions will take place:

The EnterpriseManager creates/activates a TrafficAnalyzer MC which monitors traffic on a LAN segment and measures IP traffic in terms of a selected set of application protocols. ***The TrafficAnalyzer receives from the Enterprise Manager a TaskDescription object (1). This TaskDescription "defines" the task, which means that it specifies the subnet to monitor, types of application protocols that need to be observed (e.g., HTTP, SMTP, FTP) and the monitoring duration.***

Kasteleijn at 866.

1901. Further examples include Kasteleijn at 868.

- g. *monitoring the computer network;*

1902. Kasteleijn discloses this element of claim 1, "monitoring the computer network." '730 Pat. at Cl. 1. For example:

In our scenario shown in Figure 4, the EnterpriseManager listens for "delay threshold exceeded at LAN NYC" events from other MCs in the DMF. When it receives such an event, the following set of actions will take place:

The EnterpriseManager creates/activates a ***TrafficAnalyzer MC which monitors traffic on a LAN segment and measures IP traffic*** in terms of a selected set of application protocols. The TrafficAnalyzer receives from the Enterprise Manager a TaskDescription object (1). ***This TaskDescription "defines" the task, which means that it specifies the subnet to monitor, types of application protocols that need to be observed (e.g., HTTP, SMTP, FTP) and the monitoring duration.***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Kasteleijn at 866.

1903. Further examples include Kasteleijn at 866, 867, 867-868; Kasteleijn at Fig. 4.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1904. Kasteleijn discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

Management tasks can be assigned some level of autonomy, so that functions such as discovery can *scale and adapt to the managed network* without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 860.

1905. Further examples include Kasteleijn at 861, 866-867; Kasteleijn at Fig. 4.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1906. Kasteleijn discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

Management tasks can be assigned some level of autonomy, so that functions such as discovery can *scale and adapt to the managed network* without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 860.

1907. Further examples include Kasteleijn at 861.

1908. As another example, The “delay threshold exceeded at LAN NYC” event comprises predicting a failure of a network component:

**4. Implementation Example: Enterprise Manager**

In this section, we present a management application called EnterpriseManager that runs on our DMF implementation. EnterpriseManager is a management component that runs on a DMN (DMN-ZRH), and is *used to analyze IP traffic characteristics when assigned performance thresholds are exceeded.*

Kasteleijn at 866.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1909. Further examples include Kasteleijn at 861, 866; Kasteleijn at Fig. 4.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

1910. Kasteleijn discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

Management tasks can be assigned some level of autonomy, so that functions such as discovery can *scale and adapt to the managed network* without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 860.

1911. Further examples include Kasteleijn at 861.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

1912. Kasteleijn discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

Management tasks can be assigned some level of autonomy, so that functions such as discovery can *scale and adapt to the managed network* without prior configuration information. For example, if there are other subnets attached to the discovered subnets, then the agent D can repeat the above steps (or clone itself) to collect NFS daemon data.

Kasteleijn at 860.

1913. Further examples include Kasteleijn at 861, 866-867; Kasteleijn at Fig. 4.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1914. Kasteleijn discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example, Kasteleijn discloses that an MC may request further code, MCs, or management functions provided by MCs and their associated policies, where required (*i.e.* where it cannot perform its task without these components):

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

An MC may act as a management console, *integrating management functions provided by other MCs active within the DMF*. A more sophisticated MC may carry out a longer running task such as collecting traffic statistics. Another type of MC can be an *adaptive* agent which travels to a DMN with an assigned management task, discovers the management tools available at that DMN relevant to its task, and then adapts and/or composes a solution strategy for the given environment in order to efficiently carry out its task.

Kasteleijn at 861.

1915. Further examples include Kasteleijn at 862, 863, 865, 869.

1916. To the extent this limitation is not explicitly disclosed by Kasteleijn, in my opinion, it would have been obvious to a person of ordinary skill in the art. Kasteleijn contemplates software agents (MCs) relying on code and functions from other MCs in the same runtime environment, and discloses MCs exchanging messages between one another. It would have been obvious to a person of ordinary skill in the art at the time of invention to modify Kasteleijn so that an agent may request a policy from another entity (MC, DMN, etc.) when it requires additional functionality, or when another MC reports superior results.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

1917. Kasteleijn discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

In our scenario shown in Figure 4, the EnterpriseManager listens for "delay threshold exceeded at LAN NYC" events from other MCs in the DMF. When it receives such an event, the following set of actions will take place:

The EnterpriseManager creates/activates a TrafficAnalyzer MC which monitors traffic on a LAN segment and measures IP traffic in terms of a selected set of application protocols. *The TrafficAnalyzer receives from the Enterprise Manager a TaskDescription object (1). This TaskDescription "defines" the task, which means that it specifies the subnet to monitor, types of application protocols that need to be observed (e.g., HTTP, SMTP, FTP) and the monitoring duration.*

Kasteleijn at 866.

1918. Further examples include Kasteleijn at 868.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

1919. Kasteleijn discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

Figure 1 shows an enterprise network which is partitioned into several sub-networks (subnets). We assume that all relevant devices in the system support our architecture. The goal of the example management application is to ***determine the status of all available NFS daemons running in the enterprise network***. The application, running possibly as a background troubleshooter, ***does not have a priori knowledge of the topology of the network or the location of the NFS servers***.

Kasteleijn at 858.

1920. Further examples include Kasteleijn at 859–860; Kasteleijn at Fig. 1.

- b. *constructing a topological representation of the computer network from the information.*

1921. Kasteleijn discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

Figure 1 shows an enterprise network which is partitioned into several sub-networks (subnets). We assume that all relevant devices in the system support our architecture. The goal of the example management application is to ***determine the status of all available NFS daemons running in the enterprise network***. The application, running possibly as a background troubleshooter, ***does not have a priori knowledge of the topology of the network or the location of the NFS servers***.

Kasteleijn at 858.

1922. Further examples include Kasteleijn at 859–860; Kasteleijn at Fig. 1.

**4. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

1923. Kasteleijn discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

Figure 1 shows an enterprise network which is partitioned into several sub-networks (subnets). We assume that all relevant devices in the system



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

support our architecture. The goal of the example management application is to *determine the status of all available NFS daemons running in the enterprise network*. The application, running possibly as a background troubleshooter, *does not have a priori knowledge of the topology of the network or the location of the NFS servers*.

Kasteleijn at 858.

1924. Further examples include Kasteleijn at 859–860; Kasteleijn at Fig. 1.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

1925. Kasteleijn discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6.

6. ***Claim 7***

- a. *A computer network, comprising:*

1926. Kasteleijn discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example, Kasteleijn at Figs. 2, 4 illustrates this preamble.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

1927. Kasteleijn discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

1928. Kasteleijn discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

1929. Kasteleijn discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is capable of perceiving its own state; and*

1930. Kasteleijn discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

1931. Kasteleijn discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

1932. Kasteleijn discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

## **2. Distributed Management Framework**

Distributed Management Framework (DMF) provides an object-oriented architecture where lightweight, mobile management applications can be dynamically deployed for distributed management. As shown in Figure 2, the Distributed Management Framework consists of a collection of *distributed management nodes (DMNs), where each DMN provides the execution environment and supporting services for management components (MCs), i.e., the management applications*. The DMNs form a network of peers over which hierarchies or cooperating sets of management components can be run. A *distributed directory*, maintained by each DMN is used in locating management components and services. Components and configuration of the DMF are stored in a set of replicated *code and configuration stores (CCSs)*, from where they can be downloaded to the DMNs as required. In the following, we describe these components in detail.

Kasteleijn at 860.

1933. Further examples include Kasteleijn at 861-862, 864; Kasteleijn at Figs. 2, 3.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

1934. Kasteleijn discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*of a network component*

1935. Kasteleijn discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

1936. Kasteleijn discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

1937. Kasteleijn discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

1938. Kasteleijn discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

1939. Kasteleijn discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**8. Claim 11**

- a. *The computer network of claim 7, wherein the network control*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism comprises a communications mechanism to facilitate communications with agents.*

1940. Kasteleijn discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

## **2. Distributed Management Framework**

Distributed Management Framework (DMF) provides an object-oriented architecture where lightweight, mobile management applications can be dynamically deployed for distributed management. As shown in Figure 2, the Distributed Management Framework consists of a collection of *distributed management nodes* (DMNs), where each DMN provides the execution environment and supporting services for *management components* (MCs), i.e., the management applications. ***The DMNs form a network of peers over which hierarchies or cooperating sets of management components can be run. A distributed directory, maintained by each DMN is used in locating management components and services.*** Components and configuration of the DMF are stored in a set of replicated *code and configuration stores* (CCSs), from where they can be downloaded to the DMNs as required. In the following, we describe these components in detail.

Kasteleijn at 860.

1941. Further examples include Kasteleijn at 862; Kasteleijn at Fig. 3.

## **9. Claim 12**

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

1942. Kasteleijn discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

The distributed management architecture presented in this paper provides an environment which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates. Management tasks are supported by a distributed directory and ***secure communications.***

Kasteleijn at 858.

1943. Further examples include Kasteleijn at 862, 864, 865.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

1944. Kasteleijn discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

**2. Distributed Management Framework**

Distributed Management Framework (DMF) provides an object-oriented architecture where lightweight, mobile management applications can be dynamically deployed for distributed management. As shown in Figure 2, the Distributed Management Framework consists of a collection of *distributed management nodes* (DMNs), where each DMN provides the execution environment and supporting services for *management components* (MCs), i.e., the management applications. The DMNs form a network of peers over which hierarchies or cooperating sets of management components can be run. A *distributed directory*, maintained by each DMN is used in locating management components and services. Components and configuration of the DMF are stored in a set of replicated *code and configuration stores* (CCSs), from where they can be downloaded to the DMNs as required. In the following, we describe these components in detail.

Kasteleijn at 860 (emphasis in original).

1945. Further examples include Kasteleijn at 861–862, 863, 864; Kasteleijn at Figs. 2, 3.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

1946. Kasteleijn discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

The management application begins by creating and then sending a discovery agent D to the IP router as shown in Figure 1 (1, 2).

Kasteleijn at 859.

1947. Further examples include Kasteleijn at 863, 866.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

1948. Kasteleijn discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

The management application begins by creating and then sending a discovery agent D to the IP router as shown in Figure 1 (1, 2).

Kasteleijn at 859.

1949. Further examples include Kasteleijn at 863, 866.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

1950. Kasteleijn discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

**Local Directory**

The local directory tree is ***maintained by the service manager*** and is a part of the distributed directory. It contains the following information:

DMN static configuration, e.g., the location of the distributed directory root node;

DMN capabilities, e.g., the functions such as MC types or management tools supported by this DMN; and

***DMN dynamic configuration, e.g., active MCs at this DMN and information specific to the active MCs.***

Kasteleijn at 863.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

1951. Kasteleijn discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Components and configuration of the DMF are stored in a set of replicated code and configuration stores (CCSs), from where they can be downloaded to the DMNs as required. In the following, we describe these components in detail.*

Kasteleijn at 860.

1952. Further examples include Kasteleijn at 862, 863.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

1953. Kasteleijn discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

1954. Kasteleijn discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

1955. Kasteleijn discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

1956. Kasteleijn discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 1 shows an enterprise network which is partitioned into several sub-networks (subnets). We assume that all relevant devices in the system support our architecture. ***The goal of the example management application is to determine the status of all available NFS daemons running in the enterprise network. The application, running possibly as a background troubleshooter,*** does not have a priori knowledge of the topology of the network or the location of the NFS servers.

Kasteleijn at 858.

1957. Further examples include Kasteleijn at Fig. 1; Kasteleijn at 859–860, 861.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

1958. Kasteleijn discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

The distributed management framework (DMF) presented in this paper provides an environment which allows a broad range of management tasks to move and run anywhere within the managed system. In our approach, management tasks are lightweight applications that can be dynamically configured and downloaded as required, reducing the load on managed resources and simplifying the problem of management software updates. We present an object-oriented, Java-based implementation of the DMF and describe applications developed on this platform.

Kasteleijn at 857.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

1959. Kasteleijn discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

1960. Kasteleijn discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *is capable of perceiving its own state; and*

1961. Kasteleijn discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself;*

1962. Kasteleijn discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

1963. Kasteleijn discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

1964. Kasteleijn discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

1965. Kasteleijn discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a predictive algorithm;*

1966. Kasteleijn discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

1967. Kasteleijn discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

1968. Kasteleijn discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

1969. Kasteleijn discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

1970. Kasteleijn discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

1971. Kasteleijn discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

1972. Kasteleijn discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

1973. Kasteleijn discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

1974. Kasteleijn discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

**24. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Algorithm.*

1975. Kasteleijn discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

1976. I expect to testify that Kasteleijn anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Kasteleijn discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-2.

**K. Anticipation by and/or Obviousness in View of *Proactive Management of Computer Networks using Artificial Intelligence Agents and Techniques*, by da Rocha, et al., published in the 1997 issue of Integrated Network Management V (Springer).**

1977. As explained in detail below and in the chart attached as Ex. A-6, da Rocha anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

1978. da Rocha discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

This work was developed in the area of Computer Network Management. ***The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents.*** The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

1979. Further examples include da Rocha at Sections 1, 2.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

1980. da Rocha discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

This work was developed in the area of Computer Network Management. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. *The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.*

da Rocha at Abstract.

1981. Further examples include da Rocha at 611, 613. *See generally, e.g.,* da Rocha at Section 3.

c. *is able to communicate with other software agents in the computer network;*

1982. da Rocha discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

As previously stated, the concept of proactive management involves the anticipation of possible problems which may occur in a computer network and their detection before they occur, and not merely reporting their existence. *It is fundamental that abnormal network operation be observed, that symptoms be collected and that larger problems which may come to occur be diagnosed correctly, or that anomalies be registered when it is not possible to collect enough evidence which associates an event with a known problem.* It is also necessary to maintain constant observation of the network, so that based on this knowledge enough data can be collected in order to identify what might be a symptom and relate it to a known problem. In an ideal situation, the LAN management tools establish a framework within which devices like smart hubs can monitor network activity and place information at distributed management platforms which automatically will generate trouble ticket, operational costs and usage reports [JAN 93].

da Rocha at 611-12.

1983. Further examples include da Rocha at 613.

1984. To the extent that this limitation is not expressly disclosed by da Rocha, this limitation would have been obvious to a person of ordinary skill in the art because the coordination of multiple agents implies an ability to communicate with other agents.

d. *is capable of perceiving its own state*

1985. da Rocha discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*The task of monitoring and observing the network is best achieved by agents with resident action in the machines to be managed.* In view of this, Agent 6 was created with the intent of measuring quality and operation in the network communication services by monitoring the volume of traffic and verifying the number of errors. These characteristics will later be used as a database for establishing a baseline.

da Rocha at 613.

1986. Further examples include da Rocha at Section 3.

e. *and is able to clone itself,*

1987. da Rocha discloses this element of claim 1, “and is able to clone itself.” ’730 Pat.

at Cl. 1. For example:

*Agent 6 was installed in the Darwin workstation*, which is the server for the local network and the gateway for CESUP, being monitored the ie0-bus interfaces of the UFRGS network and the ie1-bus of the CESUP local network. Agent 6 operated in two basic modes supported by the SunNet Manager (SNM) platform: data monitoring and event monitoring.

da Rocha at 614.

1988. To the extent that this limitation is not expressly disclosed by da Rocha, this limitation would have been obvious to a person of ordinary skill in the art because an agent is cloned each time it is installed another machine.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

1989. da Rocha discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

The task of monitoring and observing the network is best achieved by agents with resident action in the machines to be managed. *In view of this, Agent 6 was created with the intent of measuring quality and operation in the network communication services by monitoring the volume of traffic and verifying the number of errors.* These characteristics will later be used as a database for establishing a baseline.

da Rocha at 613.

1990. Further examples include da Rocha at Abstract. *See generally, e.g.,* da Rocha at Section 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

g. *monitoring the computer network;*

1991. da Rocha discloses this element of claim 1, “monitoring the computer network.”

’730 Pat. at Cl. 1. For example:

***The task of monitoring and observing the network is best achieved by agents with resident action in the machines to be managed.*** In view of this, Agent 6 was created with the intent of measuring quality and operation in the network communication services by monitoring the volume of traffic and verifying the number of errors. These characteristics will later be used as a database for establishing a baseline.

da Rocha at 613.

1992. Further examples include da Rocha at 613, 614, 618, 620.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

1993. da Rocha discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

As previously stated, ***the concept of proactive management involves the anticipation of possible problems which may occur in a computer network and their detection before they occur,*** and not merely reporting their existence. It is fundamental that abnormal network operation be observed, that symptoms be collected and that larger problems which may come to occur be diagnosed correctly, or that anomalies be registered when it is not possible to collect enough evidence which associates an event with a known problem. ***It is also necessary to maintain constant observation of the network, so that based on this knowledge enough data can be collected in order to identify what might be a symptom and relate it to a known problem.*** In an ideal situation, the LAN management tools establish a framework within which devices like smart hubs can monitor network activity and place information at distributed management platforms which automatically will generate trouble ticket, operational costs and usage reports [JAN 93].

da Rocha at 611-12.

1994. As another example, the inference machine can be given an objective, and models test policies to determine the likelihood of achieving its objective. The inference machine continues modelling until it finds an optimal policy to accomplish its objective. This model includes predicting a failure of the network component by the agent predicting that there is an uncorrectable fault:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The inference machine selects and applies the appropriate rule during each step in the expert system, manipulating the knowledge base. ***The inference machine can base itself on premises or elementary bits of information, and tries to achieve its objective through a combination of the two.*** In this case, it is said that it finds its way forward. The machine can also base itself on an objective and verify the needed premises using the facts involved and arrive at a conclusion. In this case, it is said that it works backwards. Inference machines that use a mixture of these approaches are those which are most successful, since, in most cases, the choices made in the inference process are reproductions of the processes a human would be likely to employ.

da Rocha at 612-13.

1995. Further examples include da Rocha at 613, 618; da Rocha at Fig. 5. *See generally*, e.g., da Rocha at Abstract; Sections 2, 3.

- i. *including predicting a failure of a network component based on a prediction algorithm*

1996. da Rocha discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

As previously stated, ***the concept of proactive management involves the anticipation of possible problems which may occur in a computer network and their detection before they occur***, and not merely reporting their existence. It is fundamental that abnormal network operation be observed, that symptoms be collected and that larger problems which may come to occur be diagnosed correctly, or that anomalies be registered when it is not possible to collect enough evidence which associates an event with a known problem. ***It is also necessary to maintain constant observation of the network, so that based on this knowledge enough data can be collected in order to identify what might be a symptom and relate it to a known problem.*** In an ideal situation, the LAN management tools establish a framework within which devices like smart hubs can monitor network activity and place information at distributed management platforms which automatically will generate trouble ticket, operational costs and usage reports [JAN 93].

da Rocha at 611-12.

1997. As another example, the inference machine can be given an objective, and models test policies to determine the likelihood of achieving its objective. The inference machine continues modelling until it finds an optimal policy to accomplish its objective. This model includes predicting a failure of the network component by the agent predicting that there is an uncorrectable fault:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The inference machine selects and applies the appropriate rule during each step in the expert system, manipulating the knowledge base. ***The inference machine can base itself on premises or elementary bits of information, and tries to achieve its objective through a combination of the two.*** In this case, it is said that it finds its way forward. The machine can also base itself on an objective and verify the needed premises using the facts involved and arrive at a conclusion. In this case, it is said that it works backwards. Inference machines that use a mixture of these approaches are those which are most successful, since, in most cases, the choices made in the inference process are reproductions of the processes a human would be likely to employ.

da Rocha at 612-13.

1998. As an additional example, agents monitor the network to gather a baseline. Then, a prediction algorithm is used to calculate thresholds for determining when an event should be reported.

These same platforms should provide services through the application of management with the capacity of configuration and planning. In Proactive Management, it is necessary to have a comparison of many management tools so that, when data is computed, a report can be created which indicates the cause to be inspected. ***The majority of these tools work with thresholds which establish the limits where the reporting of events like number of errors, specific types of packets and other parameters regarding the selection of intervals is to begin.*** It is also important to add related databases which makes for easier access and more versatile use of management data, making the integration of other tools possible.

da Rocha at 612.

1999. Further examples include da Rocha at 612, 613, 616, 620. *See generally, e.g.,* da Rocha at Abstract; Sections 2, 3.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2000. da Rocha discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

As previously stated, ***the concept of proactive management involves the anticipation of possible problems which may occur in a computer network and their detection before they occur,*** and not merely reporting their existence. It is fundamental that abnormal network operation be observed, that symptoms be collected and that larger problems which may come to occur be diagnosed correctly, or that anomalies be registered when it is not possible to collect enough evidence which associates an event with a known problem. ***It is also necessary to maintain constant observation of the network, so that based on this knowledge enough data can be collected in order to identify what might be a symptom and relate it to a known***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**problem.** In an ideal situation, the LAN management tools establish a framework within which devices like smart hubs can monitor network activity and place information at distributed management platforms which automatically will generate trouble ticket, operational costs and usage reports [JAN 93].

da Rocha at 611-12.

2001. As another example, the inference machine can be given an objective, and models test policies to determine the likelihood of achieving its objective. The inference machine continues modelling until it finds an optimal policy to accomplish its objective. This model includes predicting a failure of the network component by the agent predicting that there is an uncorrectable fault:

The inference machine selects and applies the appropriate rule during each step in the expert system, manipulating the knowledge base. ***The inference machine can base itself on premises or elementary bits of information, and tries to achieve its objective through a combination of the two.*** In this case, it is said that it finds its way forward. The machine can also base itself on an objective and verify the needed premises using the facts involved and arrive at a conclusion. In this case, it is said that it works backwards. Inference machines that use a mixture of these approaches are those which are most successful, since, in most cases, the choices made in the inference process are reproductions of the processes a human would be likely to employ.

da Rocha at 612-13.

2002. As an additional example, agents monitor the network to gather a baseline. Then, a prediction algorithm is used to calculate thresholds for determining when an event should be reported:

These same platforms should provide services through the application of management with the capacity of configuration and planning. In Proactive Management, it is necessary to have a comparison of many management tools so that, when data is computed, a report can be created which indicates the cause to be inspected. ***The majority of these tools work with thresholds which establish the limits where the reporting of events like number of errors, specific types of packets and other parameters regarding the selection of intervals is to begin.*** It is also important to add related databases which makes for easier access and more versatile use of management data, making the integration of other tools possible.

da Rocha at 612.

2003. Further examples include da Rocha at 612, 613, 616, 620, 621. *See generally, e.g.,* da Rocha at Sections 2, 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2004. da Rocha discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

This work was developed in the area of Computer Network Management. ***The work is intended to establish a strategy for the implementation of proactive management*** in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, ***and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it***, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2005. Further examples include da Rocha at 612, 613. *See generally, e.g.,* da Rocha at Abstract; Sections 2, 3.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2006. da Rocha discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

Having these data, the following workflow was determined: run agent 6, Hostmem, Hostif and Hostperf in some sub-network nodes, measuring congestion and other statistics in different hours; based on the results, establish a baseline or base with known points as a measure of standard deviation for measurements in normal situations; implant the diagnostic model, so as to activate rules each time the measurement taken is beyond the standard parameters contained in the baseline, stipulated as: for each measurement taken from the monitoring agents that arrives within reach of the timed measurement in which it occurred, a diagnostic module will be triggered to verify whether of [sic] not problems exist according to the module's rules; if the diagnostic module verifies the a problem exists which could result in a performance drop or congestion, rules will be used to determine the motives for the event; in a third moment, after verifying the previous, measures are taken to avoid the problem, ***reporting the anomalies found to the network administrator and suggesting corrective measures.***

da Rocha at 613.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2007. Further examples include da Rocha at 613, 618; da Rocha at Fig. 6. *See generally*, e.g., da Rocha at Abstract; Sections 2, 3.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2008. da Rocha discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. *See supra*, Claim Limitations 1.5, 1.11.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2009. da Rocha discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, ***for the management of networks associated with the use of agents***. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2010. Further examples include da Rocha at 611; *See generally*, e.g., da Rocha at Sections 1, 3; *see supra*, Claim Limitation 1.6.

- b. *constructing a topological representation of the computer network from the information.*

2011. da Rocha discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

***For data monitoring, the agent remained in continuous execution, making a consultation at each time interval. At the end of each consultation, the data recovered was sent to the SNM manager. This periodic sending of data permits the generation of graphs (see figure I) by***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the SNM platform*, as well as the storage of recovered data from previous consultation by the same agent, which can then generate statistics about the status of the communication system. Since the SNM platform supports the automatic generation of events like "send mail" and permits the specification of various types of thresholds, the implementation of this kind of operation by Agent 6 brought no burden to its development. To the contrary, this facility was used in sending alerts to the network administrator.

da Rocha Pages 614-15.

2012. Further examples include da Rocha at 611; da Rocha at Fig. 1; da Rocha at Abstract.

See generally, e.g., da Rocha at Sections 1, 2, 3.

4. ***Claim 4***

a. *The method of claim 1, wherein the modeling uses a numerical method.*

2013. da Rocha discloses claim 4, "[t]he method of claim 1, wherein the modeling uses a numerical method." '730 Pat. at Cl. 4. For example:

Through the measurement of normal activity within a given period of time and the identification of performance based on statistical calculations, it is possible to establish a body of data with normal function parameters which we call the baseline. ***This baseline can be used in Proactive Management by a set of functions to establish a valid statistic which characterizes the normal operation of the network during a new period of time for a specific interval***, assessing the levels of traffic at different times on different days [JAN 93].

da Rocha at 612.

2014. Further examples include da Rocha at 611, 612; da Rocha at Abstract. See also *supra*, Claim Limitations 1.7, 1.8, 3.1; see generally, e.g., da Rocha at Sections 1, 2, 3.

2015. To the extent that this limitation is not expressly disclosed by da Rocha, this Claim Limitation was obvious to a person of ordinary skill in the art because using a numerical method to make predictions based on statistics was a widely-known method at the time.

5. ***Claim 6***

a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2016. da Rocha discloses claim 6, "[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm." '730 Pat. at Cl. 6. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, ***for the management of networks associated with the use of agents***. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2017. Further examples include da Rocha at 611. *See also supra*, Claim Limitation 1.8, 1.10, 1.11; *see generally, e.g.*, da Rocha at Sections 1, 2, 3.

**6. Claim 7**

**a. A computer network, comprising:**

2018. da Rocha discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2019. Further examples include da Rocha at 613, 619. *See generally, e.g.*, da Rocha at Sections, 1, 3.

**b. a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware**

2020. da Rocha discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

2021. da Rocha discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

2022. da Rocha discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2023. da Rocha discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2024. da Rocha discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2025. da Rocha discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2026. Further examples include da Rocha at 611, 613. *See generally, e.g.,* da Rocha at Sections 1, 2.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*thereby to determine an optimal policy for the computer network*

2027. da Rocha discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2028. da Rocha discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2029. da Rocha discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2030. da Rocha discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2031. da Rocha discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*an operational characteristic of the network.*

2032. da Rocha discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2033. da Rocha discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

To reach this goal, a method was idealized where the process remains active, periodically monitoring remote systems and analyzing the results along with other information taken from the baseline, ***where the manager receives traps from agents***, and which plays a role in the diagnosis of network problems, taking action according to threshold levels ***or merely communicating a fact which has occurred***. In short, proactive operation. [ROC 94a]

da Rocha at 613.

2034. Further examples include da Rocha at Section 3.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2035. da Rocha discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

Therefore, this paper seeks to present an adopted strategy for the implementation of proactive management in computer networks. The work that motivated the elaboration of this report was driven toward the implementation of an agent and the joint utilization of network monitoring and commands of the UNIX operating system for the management of computer networks, ***where the equipment makes up the network of the Institute of Computer Sciences and of the National Supercomputing Center, including a SunNet Manager platform, Sun workstations, Sun OS 4.1 operating system compatible with UNIX 4.3 BSD, Sun OS C compiler,***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Open Windows 2.1 and PCs in a network configuration with the National Supercomputing Center were used.*

da Rocha at 611.

2036. Further examples include da Rocha at 614. *See generally, e.g.,* da Rocha at Sections 1, 3.

2037. To the extent that this limitation is not expressly disclosed by da Rocha, this Claim Limitation was obvious to a person of ordinary skill in the art because it was widely-known at the time that commercial network platforms at the time, such as the SunNet Manager, offered or were compatible with secure communications protocols.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2038. da Rocha discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

This work was developed in the area of Computer Network Management. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2039. Further examples include da Rocha at 611, 613. *See generally, e.g.,* da Rocha at Section 3.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

2040. da Rocha discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

This work was developed in the area of Computer Network Management. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2041. Further examples include da Rocha at 611, 614. *See generally, e.g.,* da Rocha at Section 3.

2042. To the extent that this limitation is not expressly disclosed by da Rocha, this Claim Limitation was obvious to a person of ordinary skill in the art because an agent running on a machine would have some of its actions controlled by the machine.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2043. da Rocha discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

Therefore, this paper seeks to present an adopted strategy for the implementation of proactive management in computer networks. The work that motivated the elaboration of this report was driven toward the implementation of an agent and the ***joint utilization of network monitoring and commands of the UNIX operating system for the management of computer networks***, where the equipment makes up the network of the Institute of Computer Sciences and of the National Supercomputing Center, including a SunNet Manager platform, Sun workstations, Sun OS 4.1 operating system compatible with UNIX 4.3 BSD, Sun OS C compiler, Open Windows 2.1 and PCs in a network configuration with the National Supercomputing Center were used.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

da Rocha at 611.

2044. Further examples include da Rocha at 613. *See generally, e.g.,* da Rocha at Sections 1, 3.

2045. To the extent that this limitation is not expressly disclosed by da Rocha, this Claim Limitation was obvious to a person of ordinary skill in the art because popular network interfaces at the time, like the interfaces mentioned in da Rocha, typically had commands to spawn, kill, or suspend processes such as agents.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2046. da Rocha discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2047. Further examples include da Rocha at 613, 619. *See generally, e.g.,* da Rocha at Sections 1, 3.

2048. Additionally each agent runtime environment must necessarily store a list of running processes; thus da Rocha satisfies this limitation under NetFuel’s application of the claims.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*database to store the policy for the agent.*

2049. da Rocha discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

The knowledge base stores the knowledge of the expert and differentiates from a conventional database in that it is active in nature, permitting updates conforming to the context. The structure of the knowledge base will depend on the type of knowledge represented. ***To have deductive knowledge, the base will usually be composed of rules. To have modeling of physical structures, causal links or interrelationship between models, the ideal structure may be a semantic network.***

da Rocha at 612.

2050. Further examples include da Rocha at Sections 2, 3.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2051. da Rocha discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

From here, it was possible to request an explanation ***the system interface is comprised of:***

***Central Window***, which is used to show diagnostics; Path Option, a simplified explanation, showing which rules were triggered, but without assembling them; Explanation Option, explains the rules used by the system; Accept Option, registers the file in the log; Exit Option, to leave the system.

If the explanation has a number of lines above that available on the monitor, exceeding the window's size, as happened in the example, it can be scrolled down.

The Accept Option registers those suggestions which have been accepted by the administrator in a log file; if he or she leaves the system without accepting any, this fact is also registered in the log file. The purpose of the log is to validate the system itself, making it possible to later investigate the situations in which the administrator did not accept the suggestions, so as to create a record which can be used by the inference machine to construct diagnostics. The automation of this process is very important, because the daily work routine of the network administrator normally does not have time to manually procure a log file, principally due to the size of these files.

da Rocha at 620

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2052. Further examples include da Rocha at Fig. 6. *See generally, e.g.,* da Rocha at Sections 3, 4.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2053. da Rocha discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2054. da Rocha discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The knowledge base stores the knowledge of the expert *and differentiates from a conventional database in that it is active in nature, permitting updates conforming to the context.* The structure of the knowledge base will depend on the type of knowledge represented. To have deductive knowledge, the base will usually be composed of rules. To have modeling of physical structures, causal links or interrelationship between models, the ideal structure may be a semantic network.

da Rocha at 612.

2055. Further examples include da Rocha at 613, 619, 620. *See generally, e.g.,* da Rocha at Sections 2, 4.

2056. As an additional example, a user can modify a default policy (which is the “new” policy).

The conversion block is responsible for receiving the monitoring and baseline files and converting them to the Prolog fact format. These facts will form the knowledge base. Based on the knowledge base, the inference engine is used for analysis of the values of the parameters and for inferring a diagnosis. This diagnosis is supported by an explanation which indicates the motive(s) for the problem, as well as suggesting possible resolutions for it. *From the interface, the network administrator receives information from the system as well as the suggestions and has the possibility of*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*expressing an opinion, agreeing or not with the diagnosis given, Beyond this, it should, in whatever situation, describe which of the approaches were followed in trying to solve the problem.*

da Rocha at 619.

2057. Further examples include da Rocha at 620. *See also supra*, Claim Limitation 1.11; *see generally, e.g.*, da Rocha at Sections 2, 4.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2058. da Rocha discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

This work was developed in the area of Computer Network Management. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. *The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it*, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2059. Further examples include da Rocha at 612, 613. *See generally, e.g.*, da Rocha at Sections 2, 3.

2060. As an additional example, the user can set different metering modes:

A expert system is divided into four distinct phases: acquisition of knowledge; knowledge base; inference machine; explanatory interface. The acquisition of knowledge is a phase involving extraction and formulation of knowledge from an expert for use in a expert system. In this process, work is performed with "knowledge engineers", technicians specialized in the job of helping experts put their knowledge into the expert system using practical rules and knowledge structuring. *As the expert puts forth his or her knowledge, the knowledge engineer represents it as a set of heuristic rules which, when coded, drive the process by a mass of information, making the process more efficient. Thus, obtaining these rules is an important step in the acquisition of knowledge.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The knowledge base stores the knowledge of the expert and differentiates from a conventional database in that it is active in nature, permitting updates conforming to the context. The structure of the knowledge base will depend on the type of knowledge represented. *To have deductive knowledge, the base will usually be composed of rules. To have modeling of physical structures, causal links or interrelationship between models, the ideal structure may be a semantic network.*

da Rocha at 612.

2061. Further examples include da Rocha at Sections 2, 3. *See also, supra*, Claim Limitation 1.11.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2062. da Rocha discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

This work was developed in the area of ***Computer Network Management***. The work is intended to establish a strategy for the implementation of proactive management in the available management environment, i.e. the National Supercomputing Center, for the management of networks associated with the use of agents. The work was motivated by a need to explore the use of agents to identify symptoms of proactive management problems which might occur in networks, and especially to recognize a problem using artificial intelligence techniques and take reactive measures to solve it, configuring a proactive management application for the prevention of problems in computer networks.

da Rocha at Abstract.

2063. Further examples include da Rocha at 613, 619. *See generally, e.g.*, da Rocha at Sections 1, 2.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2064. da Rocha discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

2065. da Rocha discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

d. *is capable of perceiving its own state; and*

2066. da Rocha discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself;*

2067. da Rocha discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2068. da Rocha discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2069. da Rocha discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2070. da Rocha discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

2071. da Rocha discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2072. da Rocha discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2073. da Rocha discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2074. da Rocha discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2075. da Rocha discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2076. da Rocha discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2077. da Rocha discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2078. da Rocha discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2079. da Rocha discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2080. da Rocha discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

2081. I expect to testify that da Rocha anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that da Rocha discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-6.

**L. Anticipation by and/or Obviousness in View of Proactive Management of Software Aging, by Castelli, et al., published in March 2001 by IBM.**

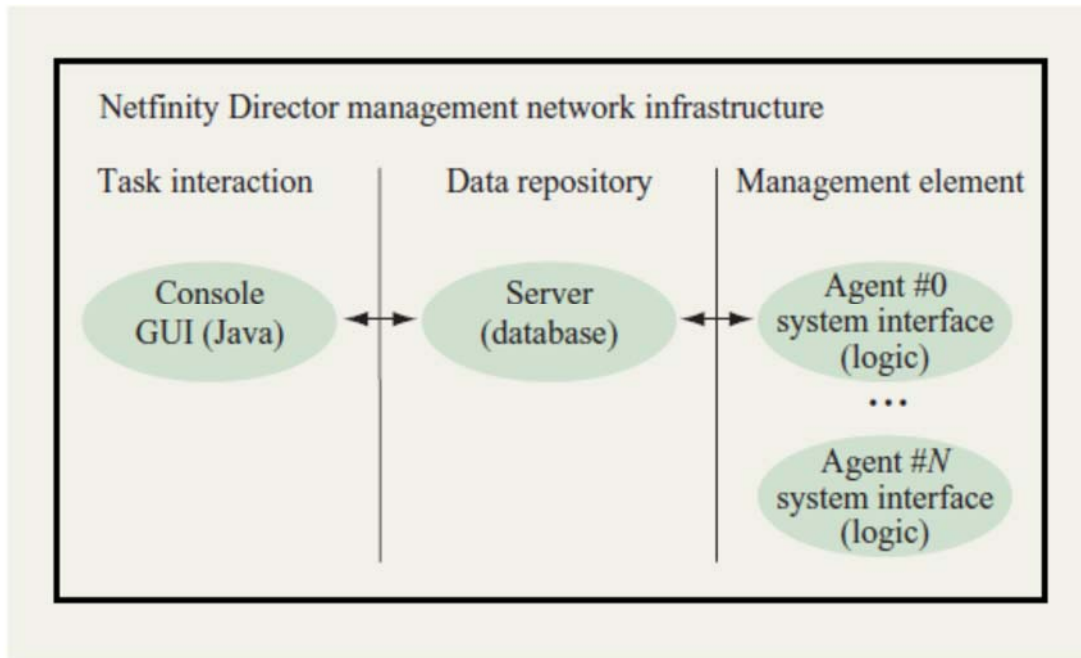
1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2082. Castelli discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

p. 313.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1**

IBM Director framework.

p. 315; *see also id.* at 311, 12, 22-23.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2083. Castelli discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

### 3. The xSeries Software Rejuvenation Agent

The xSeries Software Rejuvenation Agent (SRA) was designed to monitor consumable resources, estimate the time to exhaustion of those resources, and generate alerts to the management infrastructure when the time to exhaustion is less than a user-defined notification horizon. The management infrastructure provides a graphical user interface for the user to configure the SRA, and accepts and acts upon the alerts as described below.

The SRA was designed according to a number of ground rules, with the prime objective of maximizing flexibility, portability, and customer acceptance. For maximum generality and user acceptance, no modification to the application is allowed, to support either failure prediction or rejuvenation. Similarly, no access to the kernel is allowed, in order to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

facilitate error containment, cross-OS portability, and customer acceptance. The agent must use published and architected interfaces for data acquisition, alerting, and rejuvenation in order to minimize sensitivity to gratuitous interface churn and provide a product that is relatively stable across multiple generations of operating systems and applications. ***The agent must be relatively portable across operating systems to allow us to economically attack the different markets of commercial interest to IBM.*** A simple user interface is required which contains a minimum number of tunable parameters and is easy to set up and understand. Finally, because in many cases we do not know in advance which resources will be exhausted in the myriad environments in which we will be using SRA, the agent must be able to adapt to monitor new exhaustible parameters and execute new algorithms to predict exhaustion of those resources. These considerations caused us to partition the design into an OS-dependent data acquisition subsystem, a portable analytical subsystem with highly configurable input parameters, an architected management interface, and a management infrastructure that spans multiple IBM-supported operating systems. The SRA is incorporated as a component of IBM Director, which is discussed next.

Castelli at pp. 314-315; *see also id.* 16, 17, 18.

- c. *is able to communicate with other software agents in the computer network;*

2084. Castelli discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. All notifications are done with the IBM Director event driven notification mechanism. These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at p. 317.

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. Events provide a notification mechanism from an agent into the IBM Director environment.

Castelli at p. 315.

2085. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. *See* Ex. A-2 at limitation 1.2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *is capable of perceiving its own state*

2086. Castelli discloses this element of claim 1, “is capable of perceiving its own state.”

’730 Pat. at Cl. 1. For example:

The agents *reside on each managed system (such as an xSeries server)* and act as passive, nonintrusive *native applications*. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

Castelli at p. 315.

***Management console:***

The IBM Director management console is the graphical user interface (GUI) from which administrative tasks are performed. It is the primary interface with the administrator, and is used to configure the SRA as described below. The management console GUI is Java-based, with all state information stored on the server. *It runs as a locally installed Java application in a Java Virtual Machine (JVM\*\*).*

Castelli at p. 315.

e. *and is able to clone itself,*

2087. Castelli discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at

Cl. 1. For example:

Software rejuvenation is presented to the user as a highly stylized means to schedule rejuvenation. The user has at his disposal the building blocks of a schedule and a set of resources. The resources may be scheduled for time-based rejuvenation or configured for exhaustion forecasting.

The SRA user interface is presented to the user in the form of a calendar (Figure 4) in which the user conceptually can see and manipulate rejuvenations. To schedule a rejuvenation for a certain day, the user drags and drops a node of the cluster from the left-hand side of the interface (e.g., “platini” and “zico” are servers within cluster “copamundial”) onto the day of the week when rejuvenation is desired. A follow-up dialogue (Figure 5) then negotiates whether the user wishes the rejuvenation to occur daily, weekly, monthly, or on some other periodic basis, and the time at which the rejuvenation is to occur. The user can designate certain days of the week as invalid, when no rejuvenation can occur, and multiple rejuvenations are prohibited from being scheduled at the same time. Among other rejuvenation options, the user can engage cluster-specific logic designed to ensure that a properly planned failover can occur. As shown in the lefthand side of Figure 6, the user can ask the rejuvenation logic to confirm that at least one backup node exists which can handle the failover workload prior to rejuvenating (“check for one”), ensure that all backup nodes can handle the workload (“check for all”), or rejuvenate without checking (“skip check”). If one of the first two options is selected and a backup node cannot be found, rejuvenation is postponed until the next opportunity.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The proactive option of exhaustion prediction is provided to evoke an advanced and noninteractive scheme for scheduling resources for rejuvenation. The user can set up a stand-alone system or a clustered system for this level of support. In the expert mode of operation for exhaustion prediction, the user can allow an application to schedule rejuvenation automatically without his interaction. The user configures a cluster or a node for prediction capabilities by invoking a simple prediction configuration menu (Figure 7), and selecting the notification horizon and the type of action desired when exhaustion is predicted to occur within that horizon. Very few other parameters are configurable by the user.

Castelli at pp. 316-317.

To the extent this is not explicitly or implicitly disclosed in Castelli, it would have been obvious for a person of ordinary skill in the art at the time of invention to modify Castelli so that the SRA policy established for one node or cluster may be copied to another node/cluster, and therefore the agent responsible for that node/cluster. The motivation of this modification would be to reduce setup time and redundancy.

2088. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. See Ex. A-2 at limitation 1.2.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2089. Castelli discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

Software rejuvenation is presented to the user as a highly stylized means to schedule rejuvenation. The user has at his disposal the building blocks of a schedule and a set of resources. The resources may be scheduled for time-based rejuvenation or configured for exhaustion forecasting.

The SRA user interface is presented to the user in the form of a calendar (Figure 4) in which the user conceptually can see and manipulate rejuvenations. To schedule a rejuvenation for a certain day, the user drags and drops a node of the cluster from the left-hand side of the interface (e.g., “platini” and “zico” are servers within cluster “copamundial”) onto the day of the week when rejuvenation is desired. A follow-up dialogue (Figure 5) then negotiates whether the user wishes the rejuvenation to occur daily, weekly, monthly, or on some other periodic basis, and the time at which the rejuvenation is to occur. The user can designate certain days of the week as invalid, when no rejuvenation can occur, and multiple rejuvenations are prohibited from being scheduled at the same time. Among other rejuvenation options, the user can engage cluster-specific logic designed to ensure that a properly planned failover can occur. As shown in the lefthand side of Figure 6, the user can ask the rejuvenation logic to confirm that at least one backup node exists which can handle the failover workload prior to rejuvenating (“check for one”), ensure that all backup nodes can handle the workload (“check for all”), or rejuvenate without checking (“skip



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

check”). If one of the first two options is selected and a backup node cannot be found, rejuvenation is postponed until the next opportunity.

The proactive option of exhaustion prediction is provided to evoke an advanced and noninteractive scheme for scheduling resources for rejuvenation. The user can set up a stand-alone system or a clustered system for this level of support. In the expert mode of operation for exhaustion prediction, the user can allow an application to schedule rejuvenation automatically without his interaction. The user configures a cluster or a node for prediction capabilities by invoking a simple prediction configuration menu (Figure 7), and selecting the notification horizon and the type of action desired when exhaustion is predicted to occur within that horizon. Very few other parameters are configurable by the user.

Castelli at pp. 316-317.

g. *monitoring the computer network;*

2090. Castelli discloses this element of claim 1, “monitoring the computer network.”

’730 Pat. at Cl. 1. For example:

*IBM Director agents:*

The agents ***reside on each managed system (such as an xSeries server)*** and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

***In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes.*** These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:  
*Inventory:* IBM Director discovers new managed systems, collects the appropriate information about these systems, and stores it in the inventory database. It can then be viewed through either a default or a customized view.

*Resource monitors:* Resource monitors (**Figure 2**) enable the user to view statistics and usage of critical resources on the network. Information can be collected and monitored on attributes such as CPU, disk, memory, and network. SRA is a specialized instance of a resource monitor.  
*Event management:* Event management (**Figure 3**) enables the user to view a log of events that have occurred for a managed system or group of systems and to create event action plans to associate an event with a desired action, such as sending an e-mail, starting a program, logging to a file, or invoking rejuvenation. When the SRA has detected an impending resource exhaustion, it generates events that can be viewed using this functionality.

Castelli at pp. 315-316, *see also id.* 312, 14, 17.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the computer network,*

2091. Castelli discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

Prediction algorithms

In the current version of SRA, rejuvenation can be based on elapsed time since the last rejuvenation, or on prediction of impending exhaustion.

When using timed rejuvenation, the user interface of IBM Director is used to schedule and perform rejuvenation at a period specified by the user. A calendar interface allows the user to select when to rejuvenate different nodes of the cluster, and to select “blackout” times during which no rejuvenation is to be allowed. Although this sounds rather primitive, our analysis (presented below) shows that for typical clusters that undergo aging, system availability can be improved significantly via this technique.

Castelli at p. 317.

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

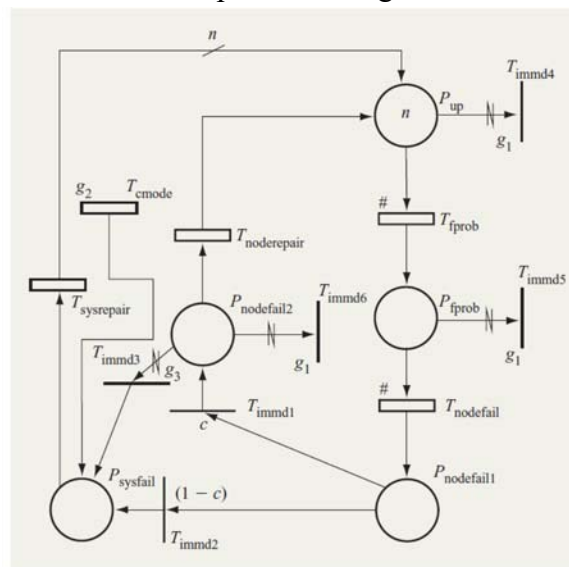
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Castelli at pp. 317-318, *see also id.* at 323, 24, 27, 28, 29.

Additionally, Castelli discloses this limitation under NetFuel's interpretation of this term. The initial parameters of the SRA agents are test policies. Execution and enforcement of these test policies then models and reflects the actual behavior of the computer network, which can then be monitored by a user or other software that can then instantiate a revised optimal policy.

- i. *including predicting a failure of a network component based on a prediction algorithm*

2092. Castelli discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:



**Figure 22**

Basic cluster system.

Castelli at p. 327

**Figure 22** shows the basic model of our cluster system. The cluster consists of  $n$  nodes. Initially, all of the nodes are in a “robust” working state, indicated by tokens in place  $P_{up}$ , in which the probability of node failure is zero. As time progresses, each node eventually transits to a “failure-probable” state (place  $P_{fprob}$ ) through the transition  $T_{fprob}$ . The nodes are still operational in this state but can fail (transit to place  $P_{nodefail1}$  with a nonzero probability). If a node crashes, it can recover with a probability  $c$  through the transition  $T_{noderepair}$ . In this case, the node goes back to the place  $P_{up}$  which represents the clean state of the node. The node recovery can fail with a probability  $(1 - c)$ , leading to a system failure (all  $n$  nodes are down). Place  $P_{sysfail}$  represents this system-level failure state.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Castelli at p. 327.

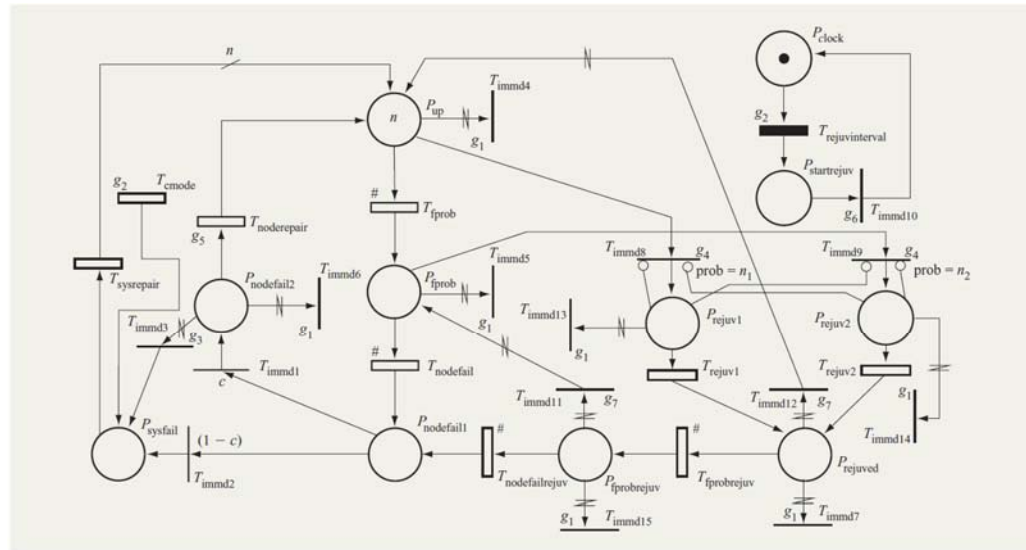


Figure 23

Cluster system employing simple time-based rejuvenation.

Castelli at p. 328, *see also id.* at 329.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2093. Castelli discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

***Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.*** The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the ***agent automatically performs the analysis.***

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and ***the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function*** without overfitting the data.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The *prediction algorithm fits several types of curves to the data in the fitting window*; these curves have been selected for their ability to capture different types of temporal trends. *A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon*. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at pp. 317-318, *see also id.* 323, 24, 27, 28, 29.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2094. Castelli discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

*Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.* The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the *agent automatically performs the analysis*.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and *the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function* without overfitting the data.

The *prediction algorithm fits several types of curves to the data in the fitting window*; these curves have been selected for their ability to capture different types of temporal trends. *A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon*. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at pp. 317-318, *see also id.* 323, 24, 27, 28, 29.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2095. Castelli discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

***Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.*** The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the ***agent automatically performs the analysis.***

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and ***the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function*** without overfitting the data.

The ***prediction algorithm fits several types of curves to the data in the fitting window***; these curves have been selected for their ability to capture different types of temporal trends. ***A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon.*** Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at pp. 317-318.

More advanced approaches are now available, such as the IBM Secureway Network Dispatcher [32]. This product provides enhanced IP-level load-balancing mechanisms and content-based routing, as well as improved management and availability functions. **Figure 17** illustrates the network dispatcher operations for a LAN implementation. ***As part of load***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*balancing, the dispatcher's scheduling policy is dynamically based on each server's load and availability. This is partly accomplished by having each server send periodic utilization information to the dispatcher.* This utilization information can easily be augmented by health information in the form of time until resource exhaustion, degree of resource exhaustion or, in its simplest form, time remaining until a timed rejuvenation. This health information can be sent to the dispatcher in order to schedule actions for individual servers. The scheduling of these actions can also take into account aggregate loading of the web host.

Castelli at p. 323; *see also id.* 324, 27, 28, 29.

2096. To the extent this limitation is not explicitly or implicitly disclosed by Castelli, it would have been obvious in view of Turek. *See* Ex. A-1 at limitation 1.11.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2097. Castelli discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

3. The xSeries Software Rejuvenation Agent

The xSeries Software Rejuvenation Agent (SRA) was designed to monitor consumable resources, estimate the time to exhaustion of those resources, and generate alerts to the management infrastructure when the time to exhaustion is less than a user-defined notification horizon. The management infrastructure provides a graphical user interface for the user to configure the SRA, and accepts and acts upon the alerts as described below.

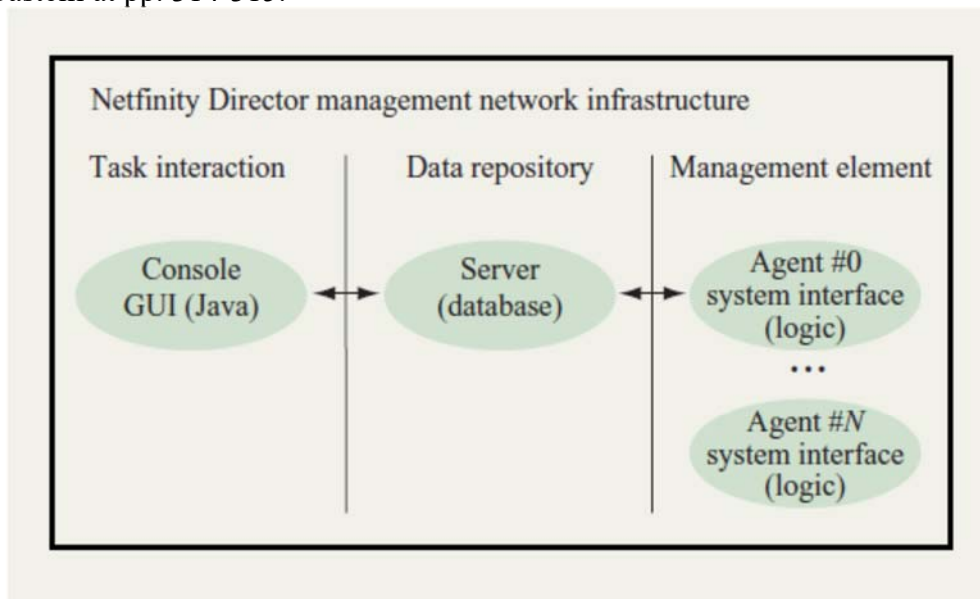
The SRA was designed according to a number of ground rules, with the prime objective of maximizing flexibility, portability, and customer acceptance. For maximum generality and user acceptance, no modification to the application is allowed, to support either failure prediction or rejuvenation. Similarly, no access to the kernel is allowed, in order to facilitate error containment, cross-OS portability, and customer acceptance. The agent must use published and architected interfaces for data acquisition, alerting, and rejuvenation in order to minimize sensitivity to gratuitous interface churn and provide a product that is relatively stable across multiple generations of operating systems and applications. ***The agent must be relatively portable across operating systems to allow us to economically attack the different markets of commercial interest to IBM.*** A simple user interface is required which contains a minimum number of tunable parameters and is easy to set up and understand. Finally, because in many cases we do not know in advance which resources will be exhausted in the myriad environments in which we will be using SRA, the agent must be able to adapt to monitor new exhaustible parameters and execute new algorithms to predict exhaustion of those resources. These considerations caused us to partition the design into an OS-dependent data acquisition subsystem, a portable analytical subsystem with highly configurable input



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

parameters, an architected management interface, and a management infrastructure that spans multiple IBM-supported operating systems. The SRA is incorporated as a component of IBM Director, which is discussed next.

Castelli at pp. 314-315.



**Figure 1**

IBM Director framework.

Castelli at p. 315; *see also id.* 316, 317, 318, 323, 24, 27, 28, 29

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2098. Castelli discloses this element of claim 3, "[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. For example:

*IBM Director agents:*

The agents ***reside on each managed system (such as an xSeries server)*** and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes.* These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:  
*Inventory:* IBM Director discovers new managed systems, collects the appropriate information about these systems, and stores it in the inventory database. It can then be viewed through either a default or a customized view.

*Resource monitors:* Resource monitors (**Figure 2**) enable the user to view statistics and usage of critical resources on the network. Information can be collected and monitored on attributes such as CPU, disk, memory, and network. SRA is a specialized instance of a resource monitor.  
*Event management:* Event management (**Figure 3**) enables the user to view a log of events that have occurred for a managed system or group of systems and to create event action plans to associate an event with a desired action, such as sending an e-mail, starting a program, logging to a file, or invoking rejuvenation. When the SRA has detected an impending resource exhaustion, it generates events that can be viewed using this functionality.

Castelli at pp. 315-316; *see id.* at 312, 14, 17.

- b. *constructing a topological representation of the computer network from the information.*

2099. Castelli discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

2100. As discussed above, it would have been obvious to combine Castelli with the OSFP routing protocol (RFC 2328). Under NetFuel’s interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. Thus it would have been obvious for SRA agents to also monitor network resources used by OSFP, and to use OSPF routing protocol. OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement Castelli’s system on network devices using OSPF.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2101. Castelli discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day's worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at pp. 317-318, *see also id.* at 323, 327, 328, 329.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2102. Castelli discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day's worth of data. The sampling

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at pp. 317-318, *see also id.* at 323, 324, 327, 328, 329.

6. ***Claim 7***

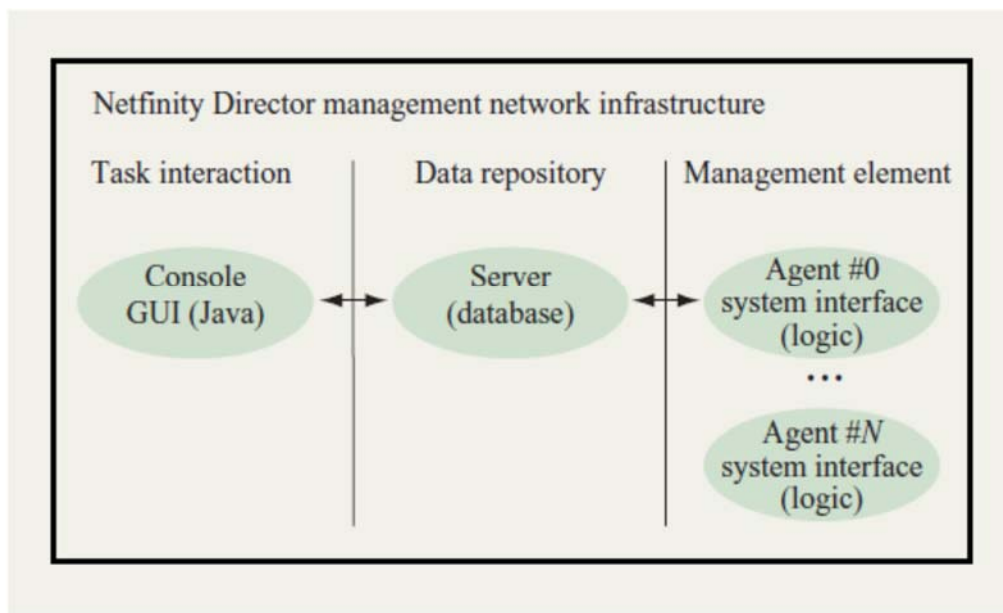
a. *A computer network, comprising:*

2103. Castelli discloses the preamble of claim 7, “[a] computer network, comprising.”

’730 Pat. at Cl. 7. For example:

The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

Castelli at p. 313.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1**

IBM Director framework.

Castelli at p. 315

***Software failure prediction and rejuvenation in a cluster environment***

A cluster is a collection of independent, self-contained computer systems working together to provide a more reliable and powerful system than a single node by itself

Castelli at p. 312

This technology has been incorporated into the IBM Director for xSeries servers.

Castelli at p. 311

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castelli at p. 323

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2104. Castelli discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

2105. Castelli discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

2106. Castelli discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

2107. Castelli discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

- f. *is able to clone itself;*

2108. Castelli discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

- g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2109. Castelli discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

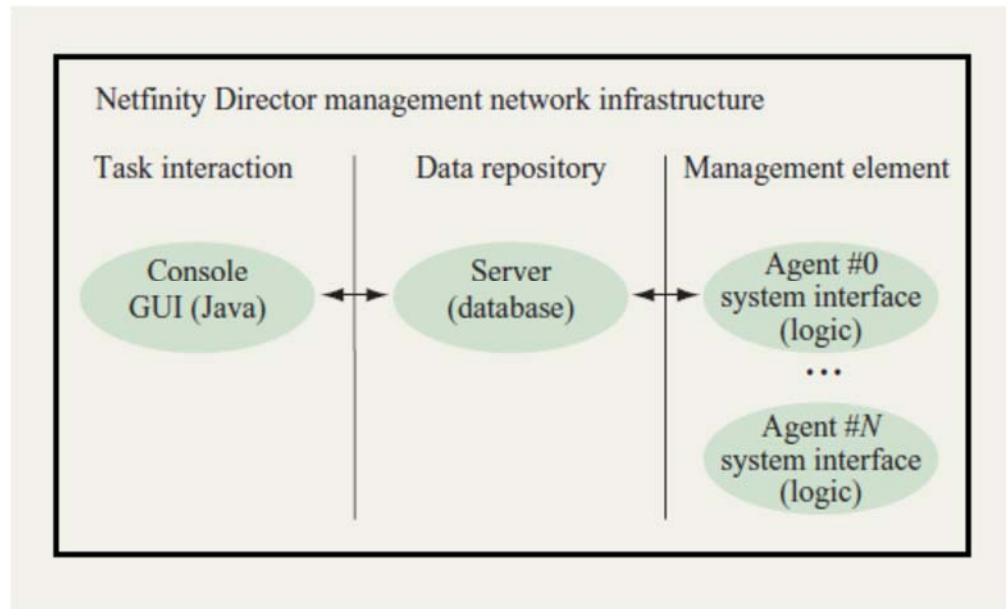
### 3. The xSeries Software Rejuvenation Agent

The xSeries Software Rejuvenation Agent (SRA) was designed to monitor consumable resources, estimate the time to exhaustion of those resources, and generate alerts to the management infrastructure when the time to exhaustion is less than a user-defined notification horizon. The management infrastructure provides a graphical user interface for the user to configure the SRA, and accepts and acts upon the alerts as described below.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The SRA was designed according to a number of ground rules, with the prime objective of maximizing flexibility, portability, and customer acceptance. For maximum generality and user acceptance, no modification to the application is allowed, to support either failure prediction or rejuvenation. Similarly, no access to the kernel is allowed, in order to facilitate error containment, cross-OS portability, and customer acceptance. The agent must use published and architected interfaces for data acquisition, alerting, and rejuvenation in order to minimize sensitivity to gratuitous interface churn and provide a product that is relatively stable across multiple generations of operating systems and applications. ***The agent must be relatively portable across operating systems to allow us to economically attack the different markets of commercial interest to IBM.*** A simple user interface is required which contains a minimum number of tunable parameters and is easy to set up and understand. Finally, because in many cases we do not know in advance which resources will be exhausted in the myriad environments in which we will be using SRA, the agent must be able to adapt to monitor new exhaustible parameters and execute new algorithms to predict exhaustion of those resources. These considerations caused us to partition the design into an OS-dependent data acquisition subsystem, a portable analytical subsystem with highly configurable input parameters, an architected management interface, and a management infrastructure that spans multiple IBM-supported operating systems. The SRA is incorporated as a component of IBM Director, which is discussed next.

Castelli at pp. 314-315.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1**

IBM Director framework.

Castelli at p. 315; *see also id.* 316, 317, 318, 323, 24, 27, 28, 29

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2110. Castelli discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2111. Castelli discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

2112. Castelli discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2113. Castelli discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2114. Castelli discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2115. Castelli discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2116. Castelli discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2117. Castelli discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at p. 317.

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. ***Events provide a notification mechanism from an agent into the IBM Director environment.***

Castelli at p. 315.

2118. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. *See* Ex. A-2 at limitation 12.0.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2119. Castelli discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. ***The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers.*** Events provide a notification mechanism from an agent into the IBM Director environment.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Castelli at p. 315.

***IBM Director agents:***

The agents *reside on each managed system (such as an xSeries server)* and act as passive, nonintrusive *native applications*. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

Castelli at p. 315.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2120. Castelli discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

2121. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. *See* Ex. A-2 at limitation 17.0.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2122. Castelli discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

2123. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. *See* Ex. A-2 at limitation 18.0.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2124. Castelli discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2125. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. See Ex. A-2 at limitation 18.0.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2126. Castelli discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

*Management server:* The management server is the platform used for the central management server, where management databases, the server engine, and management application logic reside... In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes. These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:

Castelli at p. 315.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2127. Castelli discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

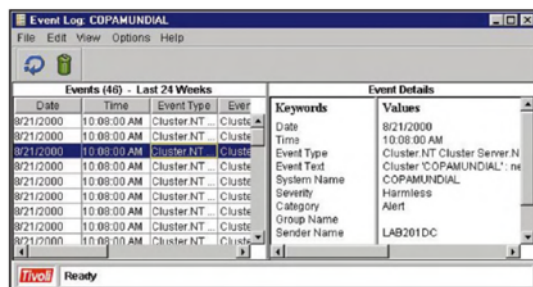
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 3

Event management.



Figure 4

Software rejuvenation calendar.

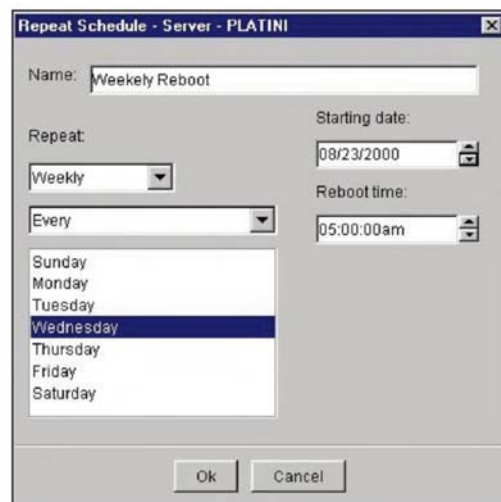


Figure 5

Scheduling rejuvenation.

Castelli at Figs. 3-5, p. 316

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Software rejuvenation is presented to the user as a highly stylized means to schedule rejuvenation. The user has at his disposal the building blocks of a schedule and a set of resources. The resources may be scheduled for time-based rejuvenation or configured for exhaustion forecasting.

The SRA user interface is presented to the user in the form of a calendar (Figure 4) in which the user conceptually can see and manipulate rejuvenations. To schedule a rejuvenation for a certain day, the user drags and drops a node of the cluster from the left-hand side of the interface (e.g., “platini” and “zico” are servers within cluster “copamundial”) onto the day of the week when rejuvenation is desired. A follow-up dialogue (Figure 5) then negotiates whether the user wishes the rejuvenation to occur daily, weekly, monthly, or on some other periodic basis, and the time at which the rejuvenation is to occur. The user can designate certain days of the week as invalid, when no rejuvenation can occur, and multiple rejuvenations are prohibited from being scheduled at the same time. Among other rejuvenation options, the user can engage cluster-specific logic designed to ensure that a properly planned failover can occur. As shown in the lefthand side of Figure 6, the user can ask the rejuvenation logic to confirm that at least one backup node exists which can handle the failover workload prior to rejuvenating (“check for one”), ensure that all backup nodes can handle the workload (“check for all”), or rejuvenate without checking (“skip check”). If one of the first two options is selected and a backup node cannot be found, rejuvenation is postponed until the next opportunity.

The proactive option of exhaustion prediction is provided to evoke an advanced and noninteractive scheme for scheduling resources for rejuvenation. The user can set up a stand-alone system or a clustered system for this level of support. In the expert mode of operation for exhaustion prediction, the user can allow an application to schedule rejuvenation automatically without his interaction. The user configures a cluster or a node for prediction capabilities by invoking a simple prediction configuration menu (Figure 7), and selecting the notification horizon and the type of action desired when exhaustion is predicted to occur within that horizon. Very few other parameters are configurable by the user.

Castelli at pp. 316-317.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2128. Castelli discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

2129. Castelli discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl.

26. For example:

2130. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Goldman. See Ex. A-3 at limitation 26.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2131. Castelli discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

Castelli discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

Management console: The IBM Director management console is the graphical user interface (GUI) from which administrative tasks are performed. It is the primary interface with the administrator, and is used to configure the SRA as described below. The management console GUI is Java-based, with all state information stored on the server. It runs as a locally installed Java application in a Java Virtual Machine (JVM\*\*). Management server: The management server is the platform used for the central management server, where management databases, the server engine, and management application logic reside.

IBM Director agents: The agents reside on each managed system (such as an xSeries server) and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

Castelli at p. 315.

This technology has been incorporated into the IBM Director for xSeries servers.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Castelli at p. 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000. Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

Castelli at p. 323.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2132. Castelli discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2133. Castelli discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2134. Castelli discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2135. Castelli discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2136. Castelli discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

2137. Castelli discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2138. Castelli discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

2139. Castelli discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2140. Castelli discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2141. Castelli discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*assigned goal of the agent is expressed as a policy.*

2142. Castelli discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

**21. Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2143. Castelli discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2144. Castelli discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2145. Castelli discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2146. Castelli discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2147. Castelli discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2148. Castelli discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

**M. Anticipation by and/or Obviousness in View of *Network Security Management with Intelligent Agents - A First Step with SLD ("Conti")*, published in the HP OpenView University Association Workshop.**

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2149. Conti discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

In this implementation we have demonstrated that ***a DIANA agent system may be easily used to monitor a network*** and perform failure recovering. As the agent are distributed, close to the NEs they have to manage and able to report only useful information, saving bandwidth consumption.

Conti at pp. 7-8.

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. The idea was to ***realise a management system*** able to continue its work even if one of its elements fails. The first

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP.

Conti at p. 5.

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.”

Conti at pp. 5-6.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2150. Conti discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

.Agents communicate between them to exchange, information called beliefs **, or objectives of management called goals**, motivations, or policies. To perform their tasks the agents need skills, which they may load from an other agent when they need them. The agents may be addressed by the users using a web browser if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.”

Conti at p. 4.

The last part is devoted to a case study implying DIANA agents managed from a web browser interface to explain the collaboration between agents.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

For this purpose, we use a scenario of fault detection, which explains the use of a System Level Diagnosis skill module. This example introduces also *how several skill modules may be used to reach a same goal*.

Conti at 1.

Reactive: generally the agent has a model of its world with *a set of predetermined actions* to perform when it receives events.

Conti at p. 2.

To perform their tasks the agents *need skills, which they may load* from an other agent when they need them.

Conti at p. 4.

The agents are hierarchically organised with the possibility *to share and delegate activities and responsibilities*. Each agent has a NM domain, which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But *an agent may owns and manage different domains* at the same time. In the same way an agent X may be the manager of an agent Y for an NM area, and Y be the manager of X for an other NM area. This may be explained by the fact that agents are collaborative and the manager role is a responsibility role.

Conti at p. 3.

When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.”

Conti at pp. 5-6.

- c. *is able to communicate with other software agents in the computer network;*

2151. Conti discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

*Agents communicate between them to exchange*, information called beliefs, or objectives of management called goals, motivations, or policies. To perform their tasks the agents need skills, which they may load from an other agent when they need them. The agents may be addressed by the users using a web browser if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.

Conti at p. 4.

The common IA properties are:

- Autonomy: does not require instruction, knows what to do in what circumstances.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.”

Conti at p.2.

Manager

Its role is to *collect and manage information reported by the management agent*, to analyse them and to take decisions.”

Conti at p. 2.

The architecture we propose is based on the extensible/flexible capabilities of IA. The agents are hierarchically organised with the possibility to *share and delegate activities* and responsibilities.

Conti at p. 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *is capable of perceiving its own state*

2152. Conti discloses this element of claim 1, “is capable of perceiving its own state.”

’730 Pat. at Cl. 1. For example:

From the previous properties are designed agent architectures, which are classically:

- Deliberative: based on a knowledge of its environment, and on reasoning capacity (logic programming).”

Conti at p. 2.

The common IA properties are:

- Autonomy: does not require instruction, knows what to do in what circumstances.

- Collaboration activity: shares work with other agents.

- Communication ability: knows how to exchange information with humans or other agents.

- Delegation: may ask on agent to do something for it.

- Deliberative: is able to reason according to its knowledge (beliefs).

Conti at p. 2.

The architecture we propose is based on the extensible/flexible capabilities of IA. The agents are hierarchically organised with the possibility to share and delegate activities and responsibilities. Each agent has a NM domain, which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But ***an agent may owns and manage different domains at the same time.***

Conti at p. 3.

When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

Conti at Fig. 2 (showing Intelligent Agent Concept)

The second one was ***a monitoring skill that may access the network elements*** through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

third developed skill module use a SLD (System Level Diagnosis) algorithm *to verify the state of a distributed system*.

Conti at p. 5.

e. *and is able to clone itself,*

2153. Conti discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at

Cl. 1. For example:

The Distributed Intelligent Agents for Network Administration (DIANA)[] project that we present, focus on the design of generic *intelligent agents able to improve their behaviour by the acquisition of new skills*.

Conti at p. 1.

We used an agent architecture of type hybrid (reactive and deliberative) keeping in mind the objective of being able to dynamically improve the behaviour of our IAs.

Several internal components are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. *Different kinds of skill may be developed and used*, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. *So when a new belief has been created*, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

The common IA properties are:

- Autonomy: does not require instruction, knows what to do in what circumstances.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask on agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: responds on event.
- Security: knows if an agent is corrupted or not.

Conti at p. 2.

2154. To the extent this limitation is not disclosed by Conti, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2155. Conti discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

Reactive: generally the agent has a model of its world with *a set of predetermined actions* to perform when it receives events.

Conti at p. 2.

Management agents are in charge of performing operation on the managed object, and to report management information to the manager entity.

Conti at p. 3.

The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module use a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This algorithm uses the beliefs resulting of the monitoring activity performed by the IAs, and requires to these IAs to have some common representative elements monitored (fig 3). The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an Interface skill composed by the skill module itself and a java applet.

Conti at p. 5.

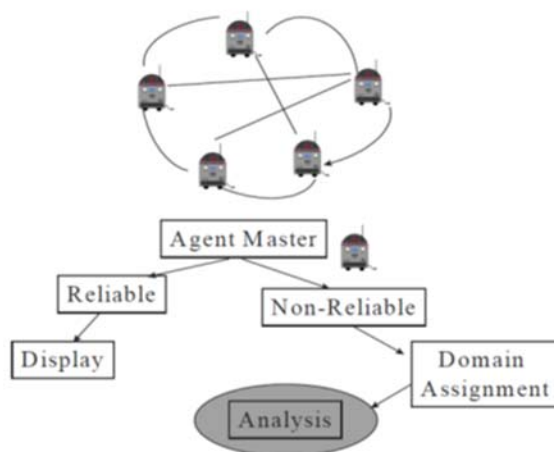
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 4

Conti at Fig. 4 (showing agent's fault analysis and modeling behavior)

2156. To the extent this limitation is not disclosed by Conti, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4.

g. *monitoring the computer network;*

2157. Conti discloses this element of claim 1, “monitoring the computer network.” ’730

Pat. at Cl. 1. For example:

The second one was *a monitoring skill that may access the network* elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module use a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This algorithm uses the beliefs resulting of *the monitoring activity performed by the IAs*, and requires to these IAs to have some common representative elements monitored (fig 3).

Conti at p. 5.



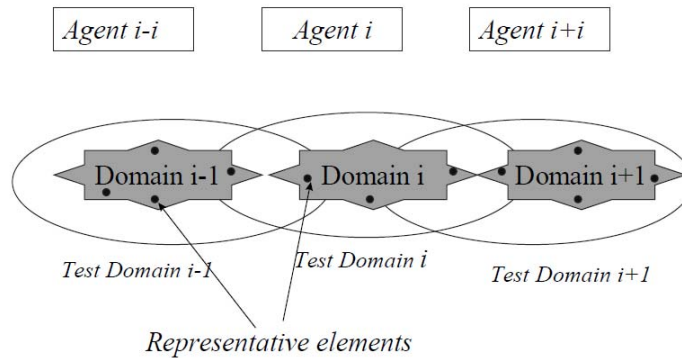
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 3

Conti at Fig. 3 (showing agents' common representative elements monitored)

The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its *monitoring activity*.

Conti at p. 7.

In this implementation we have demonstrated that a DIANA agent system may be easily used to *monitor a network* and perform failure recovering.

Conti at p. 7.

The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks *every IA to monitor their NEs* and report the results.

Conti at p. 5.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2158. Conti discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Different kinds of skill may be developed and used*, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. ***So when a new belief has been created***, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

The architecture we propose is based on the extensible/flexible capabilities of IA. The agents are hierarchically organised with the possibility to share and delegate activities and responsibilities. Each agent has a NM domain, which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or ***Fault management tasks***. But an agent may owns and manage different domains at the same time.

Conti at p. 3.

The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module use a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This algorithm uses the beliefs resulting of the monitoring activity performed by the IAs, and requires to these IAs to have some common representative elements monitored (fig 3). The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an Interface skill composed by the skill module itself and a java applet.

Conti at p. 5.

See also Conti at 2, 4, 6.

- i. *including predicting a failure of a network component based on a prediction algorithm*

2159. Conti discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

The third developed skill module use a SLD (System Level Diagnosis) ***algorithm to verify the state of a distributed system***. This algorithm uses the beliefs resulting of the monitoring activity performed by the IAs, and requires to these IAs to have some common representative elements monitored (fig 3). The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at p. 5.

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.

Conti at pp. 5-6.

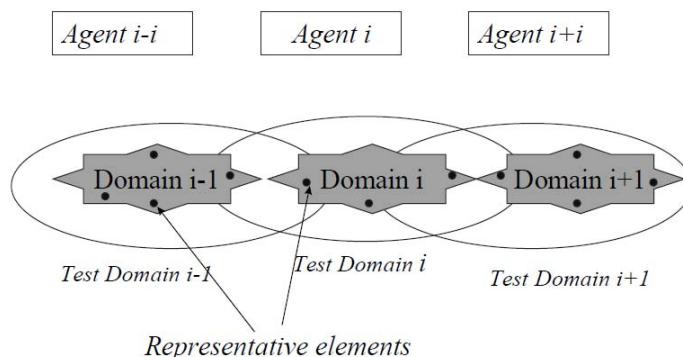


Figure 3

Conti at Fig. 3 (showing agents' common representative elements monitored)

j. *wherein said modeling comprises determining appropriate policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

2160. Conti discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet *choose and force a NE* under the responsibility of the unreliable agent *to go down*
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet *ask the master to load and start the* SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.

Conti at pp. 5-6.

At this moment *the SLD diagnoses that an agent is unreliable* and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master’s Brain to the Agent Management skill *which understand that it has to redistribute the agent domain (NEs)* of this agent to the other available agents. The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

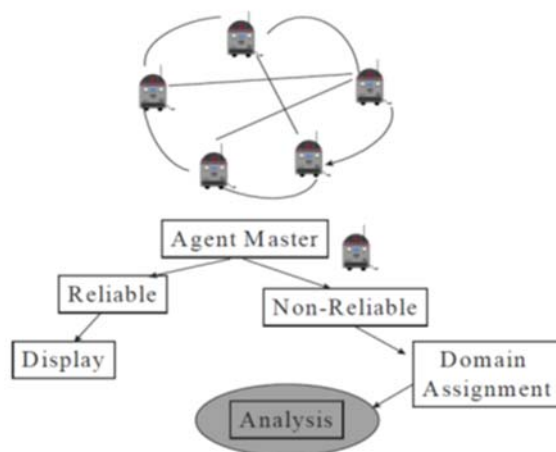
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 4

Conti at Fig. 4 (showing agent's fault analysis and modeling behavior)

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2161. Conti discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl.

1. For example:

At this moment *the SLD diagnoses that an agent is unreliable* and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master's Brain to the Agent Management skill *which understand that it has to redistribute the agent domain (NEs)* of this agent to the other available agents. The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

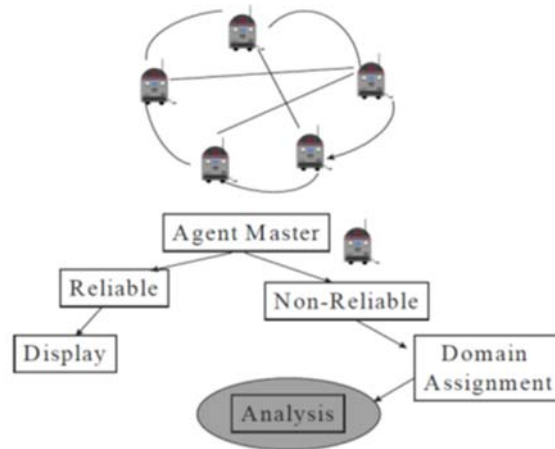
**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 4

Conti at Fig. 4 (showing agent's fault analysis and modeling behavior).

We used an agent architecture of type hybrid (reactive and deliberative) keeping in mind the objective of being able to *dynamically* improve the behaviour of our IAs.

Conti at p. 3.

The common IA properties are:

- Autonomy: does not require instruction, *knows what to do in what circumstances*.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask on agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.
- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: responds on event.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Security: knows if an agent is corrupted or not.

Conti at p. 2.

When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, *the brain know and my forward it to all interested skill modules.*

Conti at p. 4.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2162. Conti discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

Agents communicate between them to exchange, information called beliefs , or objectives of management called goals, motivations, or policies. To perform their tasks the agents need skills, which they may load from an other agent when they need them.

Conti at p. 4.

The common IA properties are:

- Autonomy: does not require instruction, knows what to do in what circumstances.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask on agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.
- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: responds on event.
- Security: knows if an agent is corrupted or not.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at p. 2.

When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

At this moment the SLD diagnoses that an agent is unreliable and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master's Brain to the Agent Management skill which understand that it has to redistribute the agent domain (NEs) of this agent to the other available agents. The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. The idea was to realise a management system able to continue its work even if one of its elements fails. The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module use a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This algorithm uses the beliefs resulting of the monitoring activity performed by the IAs, and requires to these IAs to have some common representative elements monitored (fig 3). The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results.

Conti at p. 5.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2163. Conti discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

Agents communicate between them to exchange, information called beliefs , or objectives of management called goals, motivations, or **policies**. To perform their tasks the agents need skills, which they may load from an other agent when they need them.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at p. 4.

2164. To the extent this limitation is not disclosed by Conti, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 2.0.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2165. Conti discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. The idea was to realise a management system able to continue its work even if one of its elements fails. The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was ***a monitoring skill that may access the network elements*** through the protocol SNMP. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, ***SNMP get operation on NEs and will provide beliefs on the state of these NEs.***

Conti at p. 5.

Several internals component are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills ***which may access NEs having specific protocols***, or high level skill management oriented, as for the supervising of Fault Management. When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

The common IA properties are:

- Autonomy: does not require instruction, knows what to do in what circumstances.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Delegation: may ask an agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.
- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: responds on event.
- Security: knows if an agent is corrupted or not.

Conti at p. 2

- b. *constructing a topological representation of the computer network from the information.*

2166. Conti discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

The following step will consist to improve the belief language taking in account the NM specificity, and to develop a full set of NM skills to cover all Network Management areas as Agent Master Reliable Non-Reliable Display Domain Assignment Analysis for example *a topology skill module* that will be used to analyse and localise the faults and for configuration.

Conti at pp. 7-8.

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. The idea was to realise a management system able to continue its work even if one of its elements fails. The first developed skill module was an Agent Management skill, in charge of *distributing domain of management to the available agent it knows*. The second one was a monitoring skill that *may access the network elements through the protocol SNMP*. This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs.

Conti at p. 5.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2167. Conti discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The Distributed Intelligent Agents for Network Administration (DIANA)[] project that we present, focus on the design of generic *intelligent agents able to improve their behaviour by the acquisition of new skills*.

Conti at p. 1

We used an agent architecture of type hybrid (reactive and deliberative) keeping in mind the objective of being able to dynamically improve the behaviour of our IAs.

Several internal components are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. *Different kinds of skill may be developed and used*, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain loads a skill, it gets the services the skill offers, the beliefs it uses or creates, and the prerequisite other skill it needs to execute. *So when a new belief has been created*, the brain knows and forwards it to all interested skill modules.

Conti at p. 4.

The third developed skill module uses a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This *algorithm* uses the beliefs resulting from the monitoring activity performed by the IAs, and requires these IAs to have some common representative elements monitored (fig 3).

Conti at p. 5.

The common IA properties are:

- Learning capacities: acquires and uses new knowledge.

Conti at p. 2.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2168. Conti discloses claim 6, "[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm." '730 Pat. at Cl. 6. For example:

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. *The idea was to realise a management system able to continue its work even if one of its elements fails*. The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP. This is a simple polling mechanism which will

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs. The third developed skill module use a SLD (System Level Diagnosis) algorithm to verify the state of a distributed system. This algorithm uses the beliefs resulting of the monitoring activity performed by the IAs, and requires to these IAs to have some common representative elements monitored (fig 3). The next skill developed was a general Fault Management module, which just gets the domain attributed by the Agent Management skill, and asks every IA to monitor their NEs and report the results. The last skill is an Interface skill composed by the skill module itself and a java applet. Its objective is to start the Fault Management, display the monitoring results, force an agent to report erroneous information and force an NE (through the help of the monitoring) to stop any reporting.

Conti, p. 5.

A this moment *the SLD diagnoses that an agent is unreliable* and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master's Brain to the Agent Management skill *which understand that it has to redistribute the agent domain (NEs)* of this agent to the other available agents. The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

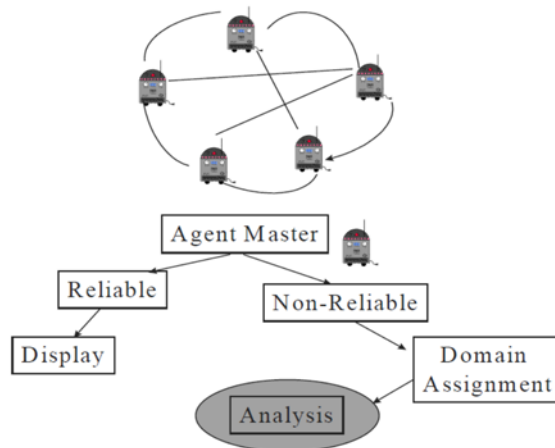


Figure 4

Conti at Fig. 4 (showing agent's fault analysis and modeling behavior).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**6. *Claim 7*a. *A computer network, comprising:*

2169. Conti discloses the preamble of claim 7, “[a] computer network, comprising.” ’730

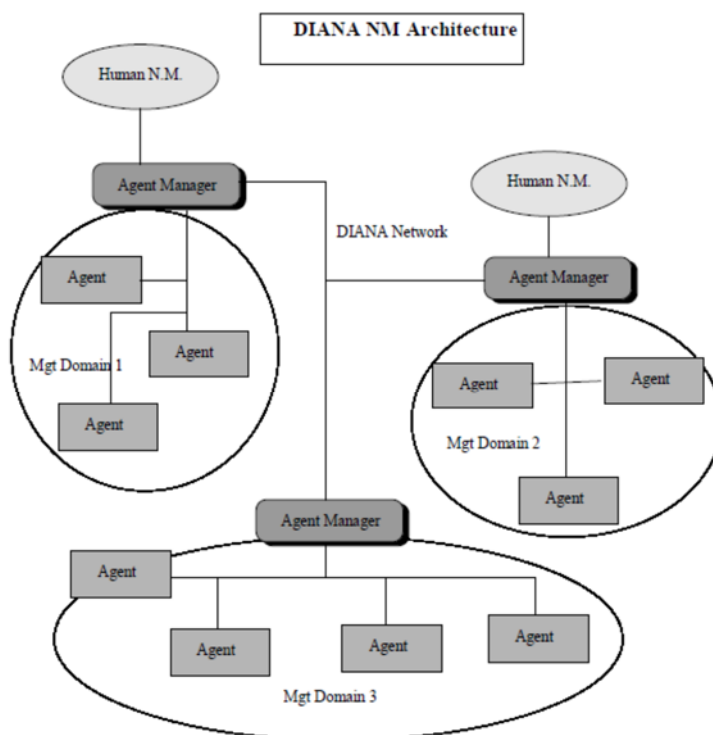
Pat. at Cl. 7. For example:

In this implementation we have demonstrated that *a DIANA agent system may be easily used to monitor a network* and perform failure recovering. As the agent are distributed, close to the NEs they have to manage and able to report only useful information, saving bandwidth consumption.

Conti at pp. 7-8.

To validate our model of IA we have developed several skill modules that are expected to be useful for a NMS. The idea was to realise a management *system* able to continue its work even if one of its elements fails. The first developed skill module was an Agent Management skill, in charge of distributing domain of management to the available agent it knows. The second one was a monitoring skill that may access the network elements through the protocol SNMP.

Conti at p. 5.



**Figure 1**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at Fig. 1 (showing DIANA NM Architecture).

Current Network Management Systems (NMS) are known as centralised systems. Some inherent known problems are lack of scalability, complexity to configure, network congestion, etc. In response to these issues, the Agent paradigm appears to be an interesting solution. The Distributed Intelligent Agents for Network Administration (DIANA)[] project that we present, *focus on the design of generic intelligent agents* able to improve their behaviour by the acquisition of new skills.”

Conti at p. 1.

Then we propose the DIANA NM architecture, which is based on a hierarchical organisation, according to definition of management domains. In the following section we give a brief presentation of the DIANA agent architecture, the main components or layers that have been designed to fulfil the requirement of genericity and flexibility, and the related concept of skill modules.

Conti at p. 1.

The architecture we propose is based on the extensible/flexible capabilities of IA. *The agents* are hierarchically organised with the possibility to share and delegate activities and responsibilities. Each agent has a *NM domain*, which represent *a set of NE for specific NM activities*. Their domains are different depending if, for example, they are doing Security or Fault management tasks.

Conti at p. 3.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2170. Conti discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

2171. Conti discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

2172. Conti discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is capable of perceiving its own state; and*

2173. Conti discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2174. Conti discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2175. Conti discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

Another advantage of using IA paradigm to develop NMS is the fault tolerance when a link goes down between the Manager and a part of the network. As the IAs are autonomous they may continue to manage the part of the network ***on which they are connected***, and they avoid the deluge of alarm events that can devour network bandwidth.

Conti at p. 3.

Each agent has a ***NM domain***, which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But an agent may owns and manage different domains at the same time.

Conti at p. 3.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2176. Conti discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2177. Conti discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2178. Conti discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2179. Conti discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2180. Conti discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2181. Conti discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**8. Claim 11**

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2182. Conti discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The architecture we propose is based on the extensible/flexible capabilities of IA. The agents are hierarchically organised with the possibility *to share and delegate activities* and responsibilities. Each agent has a *NM domain*, which represent *a set of NE for specific NM activities*. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But an agent may own and manage different domains at the same time. In the same way an agent X may be the manager of an agent Y for an NM area, and Y be the manager of X for an other NM area. This may be explained by the fact that agents are collaborative and the manager role is a responsibility role.

Conti at p. 3.

*Agents communicate between them to exchange*, information called beliefs, or objectives of management called goals, motivations, or policies. To perform their tasks the agents need skills, which they may load from an other agent when they need them. The agents may be addressed by the users using a web browser if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.

Conti at p. 4.

They observed that it is more efficient to distribute the work to small intelligent applications that share their knowledge, instead of creating important expert systems. The idea to process the data directly (or close) to the source, is now taken in account by whose study the next generation of NMS, and IA are good candidates to do this work.

Conti at p. 3.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2183. Conti discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

The common IA properties are:

- Security: knows if an agent is corrupted or not.

Conti at p. 2.

The managed objects describe the state of the network elements, and they depend on the protocol used mainly SNMP or CMIP.

Conti at p. 3.

To the extent this limitation is not disclosed by Conti, it is rendered obvious in combination with Kasteleijn.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

See Ex. A-2, limitation 12.0.

10. **Claim 13**

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

2184. Conti discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. For example:

The common IA properties are:

- Security: knows if an agent is corrupted or not.

Conti at p. 2.

The managed objects describe the state of the network elements, and they depend on the protocol used mainly SNMP or CMIP.

Conti at p. 3.

To the extent this limitation is not disclosed by Conti, it is rendered obvious in combination with Kasteleijn.

See Ex. A-2, limitation 12.0.

11. **Claim 16**

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2185. Conti discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

The agents are hierarchically organised with the possibility to share and delegate activities and responsibilities. Each agent has a NM domain, which ***represent a set of NE for specific NM activities***. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But ***an agent may owns and manage different domains*** at the same time. In the same way an agent X may be the manager of an agent Y for an NM area, and Y be the manager of X for an other NM area. This may be explained by the fact that agents are collaborative and the manager role is a responsibility role.

Conti at p. 3.

The scenario used is the following:

- Start an agent on six different systems.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.

Conti at pp. 5-6.

Another advantage of using IA paradigm to develop NMS is the fault tolerance when a link goes down between the Manager and a part of the network. As the IAs are autonomous they may continue to manage the part of the network on which they are connected, and they avoid the deluge of alarm events that can devour network bandwidth.

Conti at p. 3.

This is a simple polling mechanism which will perform on demand, and at the wish frequencies, SNMP get operation on NEs and will provide beliefs on the state of these NEs.

Conti at p. 5.

The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute.

Conti at p. 4.

12. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*environment controls an operation of the software agent based on predetermined criteria.*

2186. Conti discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.”

’730 Pat. at Cl. 17. For example:

At this moment *the SLD diagnoses that an agent is unreliable* and create a belief on the master agent expressing that an agent is unreliable. This information is forwarded by the master’s Brain to the Agent Management skill *which understand that it has to redistribute the agent domain (NEs)* of this agent to the other available agents. The Fault Management skill has been also advertised of wrong state of the agent and of the domain redistribution. Also it requires the agents to start the monitoring of their new NEs. At the end, the monitoring may continue correctly. As soon as the unreliable agent will be back in a good shape, it will take back its monitoring activity.

Conti at p. 7.

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.

Conti at pp. 5-6.

13. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*selected from the group comprising of spawn, kill and suspend.*

2187. Conti discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.
- Load and start the Fault Management activity to the previous designed master agent.
- Verify the state of all the NE
- From the applet force an agent to become unreliable
- From the applet choose and force a NE under the responsibility of the unreliable agent to go down
- Verify that the error is not reported (because the agent or the system it use has a failure)
- From the applet ask the master to load and start the SLD skill module.
- Verify that the agent is detected as in state unreliable and that its domain is redistributed.”

Conti at pp. 5-6.

To perform their tasks the agents need skills, which they may load from an other agent when they need them. The agents may be ***addressed by the users*** using a web browser if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.

Conti at p. 4.

2188. To the extent this limitation is not disclosed by Conti, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 18.0.

14. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*environment.*

2189. Conti discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

Agents communicate between them to exchange, information called beliefs, or objectives of management called goals, motivations, or policies. To perform their tasks the agents need skills, which they may load from an other agent when they need them.

Conti at p. 4.

Several internals component are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. ***When the Brain load a skill***, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

In this implementation we have demonstrated that a DIANA agent system may be easily used to monitor a network and perform failure recovering. As the agent are distributed, close to the NEs they have to manage and able to report only useful information, saving bandwidth consumption.

Conti at p. 7.

15. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2190. Conti discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

Several internals component are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. ***When the Brain load a skill***, it get the services the skill offers, the beliefs it uses or create, and the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

16. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2191. Conti discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

To perform their tasks the agents need skills, which they may load from an other agent when they need them. The agents may be addressed by the users ***using a web browser*** if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.

Conti at p. 4.

The last skill is an Interface skill composed by the skill module itself and a java applet. Its objective is to start the Fault Management, display the monitoring results, force an agent to report erroneous information and force an NE (through the help of the monitoring) to stop any reporting.

Conti at p. 5.

The scenario used is the following:

- Start an agent on six different systems.
- Request one of the agent (will be the master agent) to load and start the interface skill module.
- Start a web browser on a system and call the agent to start the applet.

Conti at p. 5.

17. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2192. Conti discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

18. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

2193. Conti discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl.

26. For example:

Several internal components are articulated around what is called the Brain. ***The Brain is in charge of analysing the interaction between the loaded skills***, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. ***Different kinds of skill may be developed and used***, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain loads a skill, it gets the services the skill offers, the beliefs it uses or creates, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain knows and may forward it to all interested skill modules.

Conti at p. 4.

The common IA properties are:

- Autonomy: does not require instruction, knows what to do in what circumstances.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask an agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organize its activity in function of priorities.
- Pro-activity: understands the cause of information and anticipates the effects.
- Reactivity: responds to an event.
- Security: knows if an agent is corrupted or not.”

Conti at p. 2.

To perform their tasks the agents need skills, which they may load from another agent when they need them. The agents ***may be addressed by the users***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*using a web browser* if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.

Conti at p. 4.

19. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2194. Conti discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

The common IA properties are:

- Autonomy: does not require instruction, knows what to do in what circumstances.
- Collaboration activity: shares work with other agents.
- Communication ability: knows how to exchange information with humans or other agents.
- Delegation: may ask on agent to do something for it.
- Deliberative: is able to reason according to its knowledge (beliefs).
- Learning capacities: acquires and uses new knowledge.
- Mobility: may move from a system to another and continues its activity.
- Planning: may organise its activity in function of priorities.
- Pro-activity: understand the cause of information and anticipate the effects.
- Reactivity: responds on event.
- Security: knows if an agent is corrupted or not.”

Conti at p. 2.

Current Network Management Systems (NMS) are known as centralised systems. Some inherent known problems are lack of scalability, complexity to configure, network congestion, etc. In response to these issues, the Agent paradigm appears to be an interesting solution. The Distributed Intelligent Agents for Network Administration (DIANA)[<sup>1</sup>] project that we present, ***focus on the design of generic intelligent agents*** able to improve their behaviour by the acquisition of new skills.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Conti at p. 1.

They observed that it is more efficient to distribute the work to small intelligent applications that share they knowledge, instead of creating important expert systems.

Conti at p. 3.

The architecture we propose is based on the extensible/flexible capabilities of IA. The agents are hierarchically organised with the possibility to share and delegate activities and responsibilities. Each agent has a NM domain, which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But an agent may owns and manage different domains at the same time.

Conti at p. 3.

To perform their tasks the agents need skills, which they may load from an other agent when they need them. The agents may be *addressed by the users* using a web browser if they have an interface skill, or with an other communication protocol as SMTP or even Telnet.

Conti at p. 4.

20. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2195. Conti discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

Several internals component are articulated around what is called the Brain. The Brain is in charge of analysing the interaction between the loaded skills, the incoming requests, planning the activity, and keeping "social" relationship with the other agents. The skill modules offer services to the agent's brain, and so to the other loaded skills. Different kinds of skill may be developed and used, as for example protocol skills which may access NEs having specific protocols, or high level skill management oriented, as for the supervising of Fault Management. When the Brain load a skill, it get the services the skill offers, the beliefs it uses or create, and the prerequisite other skill it needs to execute. So when a new belief has been created, the brain know and my forward it to all interested skill modules.

Conti at p. 4.

The architecture we propose is based on the extensible/flexible capabilities of IA. The agents are hierarchically organised with the possibility to share and delegate activities and responsibilities. Each agent has a NM domain,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

which represent a set of NE for specific NM activities. Their domains are different depending if, for example, they are doing Security or Fault management tasks. But an agent may own and manage different domains at the same time.

Conti at p. 3.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2196. Conti discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2197. Conti discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2198. Conti discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2199. Conti discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2200. Conti discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2201. Conti discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2202. Conti discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

2203. Conti discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2204. Conti discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2205. Conti discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

21. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2206. Conti discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

22. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2207. Conti discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2208. Conti discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

23. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2209. Conti discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2210. Conti discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

24. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*determining uses a numerical method.*

2211. Conti discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

25. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2212. Conti discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

**N. Anticipation by and/or Obviousness in View of *Network Security Management with Intelligent Agents* (“NSMIA”), NOMS 2000. 2000 IEEE/IFIP Network Operations and Management Symposium 'The Networked Planet: Management Beyond 2000.**

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2213. NSMIA discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

We propose a novel approach called ***IA-NSM (Intelligent Agents for Network Security Management)*** for intrusion detection using intelligent agent technology. IA-NSM provides a flexible integration of a multi-agent system in a classical ***networked environment to enhance its protection level*** against inherent attacks.

(NSMIA at p. 1).

The aim of this paper is to propose ***a multi-agent system to model the network*** security management, particularly the network intrusion detection.

(NSMIA at p. 2).

In this paper, we suggest to improve network security management by integrating DAI approach based on multi-agent system technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, capable of making various decisions with autonomy to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

(NSMIA at p. 2).

Intrusion detection is a practical approach for enhancing *the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

- the *behavior-based intrusion detection* approach, which discovers intrusive

activity by comparing the user or system behavior with a normal behavior profile;

- the *knowledge-based intrusion detection* approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database.

The *behavior-based intrusion detection* approach allows detecting unknown intrusions contrarily to the *knowledge-based intrusion detection* approach, which detects well-known intrusions.” (NSMIA at p. 4)

Giving more autonomy to the agent permits the system to react in “real time” to attacks and to take necessary actions to avoid severe consequences of the attack.

(NSMIA at p. 8).

In our architecture, we propose a hierarchical structure of autonomous agents. In the functional architecture (see Figure 1), we distinguish two: a *Manager Layer* and a *Local Layer*.

(NSMIA at p. 8).

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2214. NSMIA discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

Socialability: is the capability of *an agent to integrate itself in a large environment* populated by a society of agents with which the agent has to exchange messages *to achieve purposeful actions*. This property is satisfied even when systems have to share their knowledge and mental attitudes (beliefs, *goals*, desires, etc.).

(NSMIA at p. 7).

Moreover, multi-agent systems, as a sub-domain of DAI, are viewed as computational systems in which several autonomous and intelligent agents

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

interact and *work together in order to perform a set of tasks and to satisfy a set of goals* [1][5].

(NSMIA at p. 7).

In our work, the term intelligence is used in the sense that security network entities and especially NID components should provide reasoning capabilities, exhibit behavior autonomy, adaptability, interaction, communication and co-operation in order to reach some *intrusion detection goals*.

(NSMIA at p. 8).

The *manager layer* interacts with the *local layer* **by sending goals**, delegating specific monitoring/detection tasks and receiving pertinent reports and alarms. In each level, agents communicate and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.

(NSMIA at p. 9).

DIMA proposes three examples of modules: the *perception module* (procedural behavior), the deliberation module (knowledge-based behavior) and the communication module (which can be either procedural or knowledge-based).

The perception module manages the interactions between the agent and its environment. For example an agent perceives a list of suspicious events.

The deliberation module represents beliefs, intentions and knowledge of the agent. It is responsible 1) for generating adequate responses to the messages transmitted by the communication module, or to the changes detected by the perception module, and 2) *for achieving the agent goal(s). This goal can be the detection of a specific attack.*

(NSMIA at p. 12).

- c. *is able to communicate with other software agents in the computer network;*

2215. NSMIA discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

The DAI (Distributed Artificial Intelligence) concept [1] consists of a group of individual named agents that have distributed environments. ***Each agent cooperates and communicates with other agents.*** Combined knowledge and experience of the agent with the information coming from neighboring agents permits the agent to make the best (optimum in some sense) decision.

(NSMIA at p. 2).

The key characteristics of our architecture include autonomy, adaptability, efficiency and distribution to make the network intrusion detection more



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

flexible and less costly in term of maintenance. In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. It is based on a multi-agent system architecture (see Figure 1). It is viewed as a collection of autonomous and intelligent agents located in specific network entities named NSM hosts. ***These agents cooperate and communicate*** in order to perform intrusion detection tasks efficiently and achieve consequently better performance.

(NSMIA at p. 8).

The *Extranet Manager Agent* ***communicates*** with the *Security Policy Manager Agent*. This latter can specify new security policy, new monitoring tasks or new attacks to detect. The *EMA* is also responsible for distributing the set of *Local Agents* to each IMA.

(NSMIA at p. 9).

The *manager layer* interacts with the *local layer* by sending goals, delegating specific monitoring/detection tasks and receiving pertinent reports and alarms. ***In each level, agents communicate*** and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.” (NSMIA at p. 9)

Our agents are composed of seven modules: perception, deliberation, communication, action, interface, report; each executing a different task. The supervisor entity coordinates tasks of the different modules.

A *perception module*: that gathers all security-relevant events produced in the agent environment.

A *communication module*: that allows ***agents to communicate*** their analysis, decisions and knowledge.

(NSMIA at p. 10) .

The DIMA architecture (see Figure 3) relies on two layers: A first layer made up of interactive modules, which represent the different concurrent agent behaviors such as communicating, reasoning and perceiving. They provide the agents with some properties described in [6] such as autonomy and cooperation. For example, the communication module manages the interaction between the agent and some other agents of the system. Therefore, it is very important to make the agent cooperative.

(NSMIA at p. 12).

d. *is capable of perceiving its own state*

2216. NSMIA discloses this element of claim 1, “is capable of perceiving its own state.”

’730 Pat. at Cl. 1. For example:

Our agents are composed of seven modules: ***perception***, deliberation, communication, action, interface, report; each executing a different task. The supervisor entity coordinates tasks of the different modules.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

A ***perception module***: that gathers all security-relevant events produced in the agent environment.

A ***communication module***: that allows agents to communicate their analysis, decisions and knowledge.

(NSMIA at p. 10) .

As depicted in Figure 2, deliberation, ***perception*** and communication modules are performed respectively by the deliberation, ***perception*** and communication modules of DIMA. The interface module, action module and report generation module are three new modules, defining three new behaviors, that must be integrated in the DIMA platform, in order to be implemented.

(NSMIA at pp. 12-13).

The DIMA architecture (see Figure 3) relies on two layers: A first layer made up of interactive modules, which represent the different concurrent agent behaviors such as communicating, reasoning and ***perceiving***. They provide the agents with some properties described in [6] such as autonomy and cooperation. For example, the communication module manages the interaction between the agent and some other agents of the system. Therefore, it is very important to make the agent cooperative.

(NSMIA at p. 12).

These three modules are appropriate to our application domain. In fact, agents related to intrusion detection often own a deliberation behavior, and communication behavior and/or a ***perception*** behavior.

(NSMIA at p. 12).

Ferber [5] defines an agent as a computational or physical entity situated in an environment (either real or virtual) which is able to act in the environment, to ***perceive*** and partially to represent its environment and to communicate with other agents.

(NSMIA at p. 7).

Cognitive agents: A cognitive agent is able to find a solution for a complex problem while communicating with other agents and interacting with its knowledge base. Its main features include a high reasoning capacity, data processing, ***perception***, learning, control, communication and expertise per activity domain.

(NSMIA at, p. 7).

e. *and is able to clone itself,*

2217. NSMIA discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at

Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

To model intrusion detection, agents need to combine *cognitive abilities* (knowledge-based) to reason about complex situations, and reactive abilities (stimulus-response). So, an agent may have two kinds of behaviors: *reactive* and *cognitive* behaviors.

(NSMIA at p. 12).

Cognitive agents: A cognitive agent *is able to find a solution for a complex problem* while communicating with other agents and interacting with its knowledge base. Its main features include a high reasoning capacity, data processing, perception, learning, control, communication and expertise per activity domain.

(NSMIA at p. 7).

Combined knowledge and experience of the agent with the information coming from neighboring agents permits *the agent to make the best (optimum in some sense) decision*. In this paper, we suggest to improve network security management by integrating DAI approach based on multi-agent system technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. *These entities become more intelligent, capable of making various decisions* with autonomy to detect intrusions and to overcome their bad effects.

(NSMIA at p. 2).

To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2218. NSMIA discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

Ferber [5] defines an agent as a computational or physical entity situated in an environment (either real or virtual) which is able to act in the environment, to perceive and partially to represent its environment and to communicate with other agents. It is also driven by internal tendencies (*goals*, beliefs,...) and has an autonomous behavior which is the consequence of its perception, its representation and its interactions with the environment and with the agents.

(NSMIA at p. 7).

Autonomy: is the ability of an agent to operate *without direct intervention of humans* or other agents and to have some kind of control based on its internal state and/or external environment.

Socialability: is the capability of an agent to integrate itself in a large environment populated by a society of agents with which the agent has to exchange messages to achieve purposeful actions. This property is satisfied

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

even when systems have to share their knowledge and mental attitudes (beliefs, *goals*, desires, etc.).

Proactivity: is the ability of an agent to anticipate situations and change its course of action. It is a relevant property which occurs in network and system management in order to avoid disastrous effects on global performance. Indeed, *proactive agents are capable of exhibiting goal-direct behaviors* by taking some initiatives [6].

Reactivity: this kind of behavior means that *the agent reacts in real-time to changes* that occur in its environment.

Adaptability: is the ability of an agent to modify its behavior over time to *fulfill its problem-solving goals*.

Intelligence: the term “Intelligence” means that the agent is able to exhibit a certain level of intelligence priority, *ranging from predefined actions (planning)* up to self learning (define new actions).

(NSMIA at p. 7).

Moreover, multi-agent systems, as a sub-domain of DAI, are viewed as computational systems in which several autonomous and intelligent agents interact and work together *in order to perform a set of tasks and to satisfy a set of goals* [1][5].

(NSMIA at p. 7).

The *manager layer* interacts with the *local layer* by *sending goals*, delegating specific monitoring/detection tasks and receiving pertinent reports and alarms. In each level, agents communicate and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.

(NSMIA at p. 9).

The deliberation module represents beliefs, intentions and knowledge of the agent. It is responsible 1) for generating adequate responses to the messages transmitted by the *communication module*, or to the changes detected by the *perception module*, and 2) *for achieving the agent goal(s)*. This goal can be the detection of a specific attack.

(NSMIA at p. 12).

To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn.

See Ex. A-2, limitation 1.5.

g. *monitoring the computer network;*

2219. NSMIA discloses this element of claim 1, “monitoring the computer network.”

’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Distribution of activities: This important aspect is provided by most existing approaches. It is very important to distribute the control of security management among a number of entities that can ***monitor the network*** access at different points.

(NSMIA at p. 6).

According to the kind of data, an IDS uses to detect intrusive activity, we distinguish three types of IDS:

Host-based IDS, which are designed to ***monitor*** a single host. They use their own host Operating System's audit trail as the main source of input for detecting intrusions.

Distributed Host-based IDS, which are in charge of ***monitoring several hosts***. They perform intrusion detection using Operating System's audit trail or information from multiple monitored hosts. This information is processed on a central site.

(NSMIA at pp. 4-5).

The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control Intranet Manager Agents (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new ***monitoring tasks*** to the IMAs. The Extranet Manager Agent communicates with the Security Policy Manager Agent. This latter can specify new security policy, new monitoring tasks or new attacks to detect. The EMA is also responsible for distributing the set of Local Agents to each IMA.

The Intranet Manager Agent (IMA) manages the security of a local network. It controls the *Local Agents* and analyzes the ***monitored events*** reported by these agents.

(NSMIA at p. 9).

The *Local Layer* manages the security of a domain, which is constituted by a set of hosts. This layer is composed of a group of *Local agents*, which have specific functions. In fact, the *Manager Layer* specifies to the *Local Layer* ***the activities that must be monitored***. These activities can be classified in *Extranet*, *Intranet* and *Local* activities.

(NSMIA at p. 9).

The *manager layer* interacts with the *local layer* by sending goals, delegating specific ***monitoring/detection tasks*** and receiving pertinent reports and alarms. In each level, agents communicate and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.

(NSMIA at p. 9).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2220. NSMIA discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

The Extranet Manager Agent acts as a global modeler as it manages the entire network:

- he Manager Layer manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: Security Policy Manager Agent, Extranet Manager Agent and Intranet Manager Agent.

- The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.

- The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control Intranet Manager Agents (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to ***perform another analysis on suspicious events*** in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The Extranet Manager Agent communicates with the Security Policy Manager Agent. ***This latter can specify new security policy, new monitoring tasks or new attacks to detect.*** The EMA is also responsible for distributing the set of *Local Agents* to each IMA.

- The Intranet Manager Agent (IMA) manages the security of a local network. It controls the *Local Agents* and analyzes the monitored events reported by these agents.

(NSMIA at p. 9).

Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with the information coming from neighboring agents permits ***the agent to make the best (optimum in some sense) decision.***

(NSMIA at p. 2).

We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, ***capable of making various decisions*** with autonomy to detect intrusions and to overcome their bad effects.

(NSMIA at p. 2).

Administrators do not have to concern about all the security problems. They interact with the agent from a high level using security policies. Security

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

policies tell the agents what behavior they should exhibit when attacks occur. Hence, communications between agents permit to collect information. This information *permits the agents to identify attacks* that can not be detected if it is static. Giving more autonomy to the agent permits the system to react in “real time” to attacks and *to take necessary actions* to avoid severe consequences of the attack.

(NSMIA at p. 8).

- i. *including predicting a failure of a network component based on a prediction algorithm*

2221. NSMIA discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, *capable of making various decisions* with autonomy to detect intrusions and to overcome their bad effects.

(NSMIA at p. 2).

Intrusion detection is a practical approach for enhancing the security of computer and network systems. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

- the behavior-based intrusion detection approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;
- the knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user’s session and known pattern attacks stored in a database.

(NSMIA at p. 4).

Proactivity: is the ability of *an agent to anticipate situations and change its course of action*. It is a relevant property which occurs in network and system management in order to avoid disastrous effects on global performance. Indeed, proactive agents are capable of exhibiting goal-direct behaviors by taking some initiatives [6].

Reactivity: this kind of behavior means that *the agent reacts in real-time to changes* that occur in its environment.

Adaptability: is the ability of an agent to modify its behavior over time to fulfill its problem-solving goals.

(NSMIA at p. 7).

Ferber [5] defines an agent as a computational or physical entity situated in an environment (either real or virtual) which is able to act in the



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

environment, to perceive and partially to represent its environment and to communicate with other agents.

(NSMIA at p. 7).

To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.8. NSMIA discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman.

Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine NSMIA and Hellerstein to implement these models.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2222. NSMIA discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

The *Manager Layer* manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: *Security Policy Manager Agent*, *Extranet Manager Agent* and *Intranet Manager Agent*.

- The ***Security Policy Manager Agent (SPMA)*** manages the security policies specified by the security officer.

- The ***Extranet Manager Agent (EMA)*** manages the security of the entire distributed network. Its role is to manage and control *Intranet Manager Agents* (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to ***perform another analysis on suspicious events*** in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The *Extranet Manager Agent* communicates with the *Security Policy Manager Agent*. ***This latter can specify new security policy, new monitoring tasks***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*or new attacks to detect.* The *EMA* is also responsible for distributing the set of *Local Agents* to each IMA.

- The *Intranet Manager Agent (IMA)* manages the security of a local network. It controls the *Local Agents* and analyzes the monitored events reported by these agents.

(NSMIA at p. 9).

Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with the information coming from neighboring agents permits *the agent to make the best (optimum in some sense) decision.*

(NSMIA at p. 2).

We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, *capable of making various decisions with autonomy to detect intrusions and to overcome their bad effects.*

(NSMIA at p. 2).

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2223. NSMIA discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

An agent is a *pro-active entity*. It does not simply act in response to the received messages from the other agents. For example, it interacts with its environment and deliberates to *determine the most appropriate action.*

(NSMIA at p. 11).

The *deliberation module* represents beliefs, intentions and knowledge of the agent. It is responsible 1) *for generating adequate responses to the messages* transmitted by the *communication module*, or to the changes detected by the *perception module*, and 2) *for achieving the agent goal(s).* This goal can be the detection of a specific attack.

(NSMIA at p. 12).

2224. To the extent this limitation is not met, it would have been obvious in view of Goldman. See Ex. A-3, limitation 1.10. NSMIA discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman.

2225. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine NSMIA and Hellerstein to implement these models and dynamically update policies based on them.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2226. NSMIA discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

Moreover, multi-agent systems, as a sub-domain of DAI, are viewed as computational systems in which several ***autonomous and intelligent agents interact and work together in order to perform a set of tasks*** and to satisfy a set of goals [1][5].

(NSMIA at p. 7).

In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. It is based on a multi-agent system architecture (see Figure 1). It is viewed ***as a collection of autonomous and intelligent agents*** located in specific network entities named NSM hosts. These agents cooperate and communicate in order to perform intrusion detection tasks efficiently and achieve consequently better performance. In fact, by giving more autonomy to agent in the control of the overall intrusion detection, the task of administration becomes easier. Administrators do not have to concern about all the security problems. They interact with the agent from a high level using security policies. Security policies tell the agents what behavior they should exhibit when attacks occur. Hence, communications between agents permit to collect information. ***This***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*information permits the agents to identify attacks* that can not be detected if it is static. Giving more autonomy to the agent permits the system to react in “real time” to attacks and to take necessary actions to avoid severe consequences of the attack.

(NSMIA at p. 8).

The *communication module* manages the interactions between the agent and the other agents of its group(s), no matter what machine they are running on. It defines the mailbox of the agent and the way the messages are received and enqueued for later interpretation. An agent may need some others information to refine its analysis. In this case, *it asks other agents to give it the necessary information*.

(NSMIA at p. 12).

Autonomy: *is the ability of an agent to operate without direct intervention of humans or other agents* and to have some kind of control based on its internal state and/or external environment.

- Socialability: is the capability of an agent to integrate itself in a large environment populated by a society of agents with which the agent has to exchange messages to achieve purposeful actions. This property is satisfied even when systems have to share their knowledge and mental attitudes (beliefs, goals, desires, etc.).

- Proactivity: is the ability of an agent to anticipate situations and change its course of action. It is a relevant property which occurs in network and system management in order to avoid disastrous effects on global performance. Indeed, *proactive agents are capable of exhibiting goal-direct behaviors by taking some initiatives* [6].

- Reactivity: this kind of behavior means that the agent reacts in real-time to changes that occur in its environment.

- Adaptability: is the ability of an agent to modify its behavior over time to fulfill its problem-solving goals.

- Intelligence: the term “Intelligence” means that the agent is able to exhibit a certain level of intelligence priority, ranging from predefined actions (planning) up to self learning (define new actions).

(NSMIA at p. 7).

## 2. *Claim 2*

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2227. NSMIA discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The Extranet Manager Agent acts as a global modeler as it manages the entire network:

- The Manager Layer manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: Security Policy Manager Agent, Extranet Manager Agent and Intranet Manager Agent.
- The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.
- The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control *Intranet Manager Agents* (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The *Extranet Manager Agent* communicates with the *Security Policy Manager Agent*. ***This latter can specify new security policy, new monitoring tasks or new attacks to detect.*** The *EMA* is also responsible for distributing the set of *Local Agents* to each IMA.

(NSMIA at p. 9).

In fact, by giving more autonomy to agent in the control of the overall intrusion detection, the task of administration becomes easier. Administrators do not have to concern about all the security problems. They interact with the agent from a high level using *security policies*. ***Security policies tell the agents what behavior they should exhibit*** when attacks occur. Hence, communications between agents permit to collect information.

(NSMIA at p. 8).

To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 2.0.

### 3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2228. NSMIA discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

The ***communication module*** manages the interactions between the agent and the other agents of its group(s), no matter what machine they are running on. It defines the mailbox of the agent and the way the messages are received and enqueued for later interpretation. An agent may need some

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

others information to refine its analysis. In this case, *it asks other agents to give it the necessary information.*

(NSMIA at p. 12).

An **interface module**: interacts with the security administrator receiving administrator requests/specifications, delivering reports, sending alarms when an attack is detected and **asking for additional information** or confirmation when necessary. For example, the administrator can ask for the current network security status. This module exists only in the SPMA.

(NSMIA at p. 11).

Distributed Host-based IDS, which are in charge of monitoring several hosts. They perform intrusion detection using Operating System's audit trail or information from multiple monitored hosts. This information is processed on a central site.

(NSMIA at p. 5).

The DAI (Distributed Artificial Intelligence) concept [1] consists of a group of individual named agents that have distributed environments. Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with **the information coming from neighboring agents permits the agent to make the best (optimum in some sense) decision.**

(NSMIA at p. 2).

An agent is a pro-active entity. It does not simply act in response to the received messages from the other agents. For example, it interacts with its environment and deliberates to determine the most appropriate action.

(NSMIA at p. 11).

- b. *constructing a topological representation of the computer network from the information.*

2229. NSMIA discloses this element of claim 3, "constructing a topological representation of the computer network from the information." '730 Pat. at Cl. 3. For example:

Our agents are composed of seven modules: perception, deliberation, communication, action, interface, **report**; each executing a different task. The supervisor entity coordinates tasks of the different modules.

A perception module: that gathers all security-relevant events produced in the agent environment.

A communication module: that allows agents to communicate their analysis, decisions and knowledge.

An action module: its role is to take appropriate actions when an intruder is detected.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

A report generation: establishes reports on detected attacks to be sent to the administrator.

(NSMIA at p. 11).

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2230. NSMIA discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

To model intrusion detection, agents need to combine cognitive abilities (knowledge-based) to ***reason about complex situations***, and reactive abilities (stimulus-response).

(NSMIA at p. 12).

In this paper, we suggest to improve network security management by integrating DAI approach based on multi-agent system technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, capable of making various decisions with autonomy to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

(NSMIA at p. 2).

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2231. NSMIA discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

(NSMIA at Fig. 5 (showing an example of ATN of an Extranet Local Agent))

Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with the information coming from neighboring agents permits the agent to make the best (optimum in some sense) decision. In this paper, we suggest to improve network security management by integrating DAI approach based on multi-agent system technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. ***These entities become more intelligent, capable of making various decisions with autonomy*** to detect intrusions and to overcome their bad effects. The introduction of multi-

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

(NSMIA at p. 2).

Cognitive agents: A cognitive agent is *able to find a solution for a complex problem* while communicating with other agents and interacting with its knowledge base. Its main features include a high reasoning capacity, data processing, perception, learning, control, communication and expertise per activity domain.

(NSMIA at p. 7).

In fact, by giving more autonomy to agent in the control of the overall intrusion detection, the task of administration becomes easier. *Administrators do not have to concern* about all the security problems. They interact with the agent from a high level using security policies. Security policies tell the agents what behavior they should exhibit when attacks occur. Hence, communications between agents permit to collect information. This information permits the agents to identify attacks that can not be detected if it is static. *Giving more autonomy to the agent permits the system to react in “real time”* to attacks and to take necessary actions to avoid severe consequences of the attack.

(NSMIA at p. 8).

6. ***Claim 7***

a. *A computer network, comprising:*

2232. NSMIA discloses the preamble of claim 7, “[a] computer network, comprising.”

’730 Pat. at Cl. 7. For example:

The aim of this paper is to propose *a multi-agent system* to model the network security management, particularly the network intrusion detection.

(NSMIA at p. 2).

Multi-Agent Systems technology can be useful for efficiently designing and maintaining secure *networks*. Indeed, networks evolve at a rapid pace in terms of the number and type of components and user access queries as well as intrusion possibilities. Features such as autonomy, adaptability and flexibility of the “intelligent” agent paradigm allow managing network evolution in a controlled way. The focus of our work concerns one critical security management issue that is *intrusion detection*. We propose a novel approach called IA-NSM (Intelligent Agents for Network Security Management) for intrusion detection using intelligent agent technology. IA-NSM provides a flexible integration of *a multi-agent system* in a classical networked environment to enhance its protection level against inherent attacks.

(NSMIA at p.1).



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In this paper, we suggest to improve network security management by integrating DAI approach based on multi-agent system technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, capable of making various decisions with autonomy to detect intrusions and to overcome their bad effects. The introduction of multi-agent system (MAS) in a network seems so promising to embed adaptive features thereby enabling network entities to perform adaptive behavior and becoming “intelligent”.

(NSMIA at p. 2).

The key characteristics of our architecture include autonomy, adaptability, efficiency and distribution to make the network intrusion detection more flexible and less costly in term of maintenance. In our proposed approach, we define a new architecture, called IA-NSM, which supports NSM activities. It is based on a multi-agent system architecture (see Figure 1). It is viewed as a collection of autonomous and intelligent agents located in specific network entities named NSM hosts. These agents cooperate and communicate in order to perform intrusion detection tasks efficiently and achieve consequently better performance.

(NSMIA at p. 8).

Intrusion detection is a practical approach for enhancing *the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

the *behavior-based intrusion detection* approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;

te *knowledge-based intrusion detection* approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database.

te *behavior-based intrusion detection* approach allows detecting unknown intrusions contrarily to the *knowledge-based intrusion detection* approach, which detects well-known intrusions.

(NSMIA at p. 4).

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2233. NSMIA discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

2234. NSMIA discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

2235. NSMIA discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2236. NSMIA discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2237. NSMIA discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2238. NSMIA discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2239. NSMIA discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2240. NSMIA discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2241. NSMIA discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2242. NSMIA discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2243. NSMIA discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2244. NSMIA discloses claim 10, “[t]he computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**8. Claim 11**

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2245. NSMIA discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The *manager layer* interacts with the *local layer* by sending goals, delegating specific monitoring/detection tasks and receiving pertinent reports and alarms. ***In each level, agents communicate*** and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.

(NSMIA at p. 9).

The DIMA architecture (see Figure 3) relies on two layers: ***A first layer made up of interactive modules***, which represent the different concurrent agent behaviors such as ***communicating***, reasoning and perceiving. They provide the agents with some properties described in [6] such as autonomy and cooperation. For example, the ***communication module*** manages the interaction between the agent and some other agents of the system. Therefore, it is very important to make the agent cooperative.

(NSMIA at p. 12).

**Figure 3:** DIMA agent architecture

(NSMIA at Fig. 3 (showing DIMA agent architecture))

Our agents are composed of seven modules: perception, deliberation, communication, action, interface, report; each executing a different task. The supervisor entity coordinates tasks of the different modules.

*A perception module:* that gathers all security-relevant events produced in the agent environment.

*A communication module:* that allows ***agents to communicate*** their analysis, decisions and knowledge.

(NSMIA at p. 10).

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2246. NSMIA discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

They interact with the agent from a high level using security policies. ***Security policies*** tell the agents what behavior they should exhibit when attacks occur. Hence, ***communications between agents*** permit to collect information. This information permits the agents to identify attacks that can not be detected if it is static.

(NSMIA at p. 8).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The *Manager Layer* manages the global security of a network. This network can be local or distributed. In this layer we identify three levels of agents: *Security Policy Manager Agent*, *Extranet Manager Agent* and *Intranet Manager Agent*.

- The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.

- The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control *Intranet Manager Agents* (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The *Extranet Manager Agent* communicates with the *Security Policy Manager Agent*. This latter can specify new security policy, new monitoring tasks or new attacks to detect. The *EMA* is also responsible for distributing the set of *Local Agents* to each IMA.

- The Intranet Manager Agent (IMA) manages the security of a local network. It controls the *Local Agents* and analyzes the monitored events reported by these agents.

- The *Local Layer* manages the security of a domain, which is constituted by a set of hosts. This layer is composed of a group of *Local agents*, which have specific functions. In fact, the *Manager Layer* specifies to the *Local Layer* the activities that must be monitored. These activities can be classified in *Extranet*, *Intranet* and *Local* activities. According to this classification, we distinguish 3 kinds of *Local Agents*: *Extranet Local Agent*, *Intranet Local Agent* and *Internal Local Agent*.

(NSMIA at p. 9).

2247. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 12.0.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2248. NSMIA discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

The DAI (Distributed Artificial Intelligence) concept [1] consists of a group of individual named ***agents that have distributed environments***. Each agent cooperates and communicates with other agents.

(NSMIA at p. 2).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The introduction of multi-agent system (MAS) in a network seems so promising to *embed adaptive features thereby enabling network entities to perform adaptive behavior* and becoming “intelligent”.

(NSMIA at p. 2).

According to the kind of data, an IDS uses to detect intrusive activity, we distinguish three types of IDS:

*Host-based IDS, which are designed to monitor a single host.* They use their own host Operating System’s audit trail as the main source of input for detecting intrusions.

*Distributed Host-based IDS, which are in charge of monitoring several hosts.* They perform intrusion detection using Operating System’s audit trail or information from multiple monitored hosts. This information is processed on a central site.

*Network-based IDS, which analyze traffic on a LAN to detect intrusive behavior.*

(NSMIA at p. 4-5).

11. **Claim 17**

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2249. NSMIA discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.”

’730 Pat. at Cl. 17. For example:

The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

the *behavior-based intrusion detection* approach, which discovers intrusive activity by *comparing the user or system behavior with a normal behavior profile*;

the *knowledge-based intrusion detection* approach, which detects intrusions upon a comparison between parameters of the user’s session and known *pattern attacks stored in a database*.

(NSMIA at p. 4).

12. **Claim 18**

- a. *The computer network of claim 17, wherein the operation is*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*selected from the group comprising of spawn, kill and suspend.*

2250. NSMIA discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

*An interface module: interacts with the security administrator receiving administrator requests/specifications, delivering reports, sending alarms when an attack is detected and asking for additional information or confirmation when necessary. For example, the administrator can ask for the current network security status. This module exists only in the SPMA.*

(NSMIA at pp. 10-11).

2251. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 18.0.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2252. NSMIA discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

*The manager layer interacts with the local layer by sending goals, delegating specific monitoring/detection tasks and receiving pertinent reports and alarms. In each level, agents communicate and exchange their knowledge and analysis for detecting intrusive activities in a co-operative manner.*

(NSMIA at p. 9).

*A supervision module coordinates interactions between the different modules using a finite state automaton.*

(NSMIA at p. 11).

2253. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 19.0. It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

14. **Claim 21**

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2254. NSMIA discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

A ***deliberation module***: that enables agent intelligence and autonomy. The hybrid agent should be able to reason and extrapolate by relying on ***built knowledge*** and experience in a rational way. Decisions of the agent depend on the security environment status, the neighboring system evolution and its mental attitudes.

(NSMIA at p. 11).

The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

the ***behavior-based intrusion detection*** approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;

the ***knowledge-based intrusion detection*** approach, which detects intrusions upon a comparison between parameters of the user’s session and known ***pattern attacks stored in a database***.”

(NSMIA at p. 4).

To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 21.0 It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

15. **Claim 22**

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2255. NSMIA discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

Our agents are composed of seven modules: perception, deliberation, communication, action, ***interface***, report; each executing a different task. The supervisor entity coordinates tasks of the different modules.

A perception module: that gathers all security-relevant events produced in the agent environment.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

A communication module: that allows agents to communicate their analysis, decisions and knowledge.

An action module: its role is to take appropriate actions when an intruder is detected.

A report generation: establishes reports on detected attacks to be sent to the administrator.

A deliberation module: that enables agent intelligence and autonomy. The hybrid agent should be able to reason and extrapolate by relying on built knowledge and experience in a rational way. Decisions of the agent depend on the security environment status, the neighboring system evolution and its mental attitudes.

An *interface module: interacts with the security administrator receiving administrator requests/specifications*, delivering reports, sending alarms when an attack is detected and asking for additional information or confirmation when necessary. For example, the administrator can ask for the current network security status. This module exists only in the SPMA.

(NSMIA at pp. 10-11).

2256. To the extent this limitation is not met, it would have been obvious in view of Turek. See Ex. A-1 at limitation 22.0. A person of ordinary skill in the art would understand that network management tools may benefit from a GUI.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2257. NSMIA discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2258. NSMIA discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl.

26. For example:

The Security Policy Manager Agent (SPMA) manages the security policies specified by the security officer.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The Extranet Manager Agent (EMA) manages the security of the entire distributed network. Its role is to manage and control *Intranet Manager Agents* (IMA). These agents report pertinent analysis to the EMA. The role of the latter is then to perform another analysis on suspicious events in order to confirm or not the detection of an attack. It can also ask for another data processing and delegate then new monitoring tasks to the IMAs. The *Extranet Manager Agent* communicates with the *Security Policy Manager Agent*. ***This latter can specify new security policy, new monitoring tasks or new attacks to detect.*** The *EMA* is also responsible for distributing the set of *Local Agents* to each IMA.

(NSMIA at p. 9).

Each agent cooperates and communicates with other agents. Combined knowledge and experience of the agent with the information coming from neighboring agents permits ***the agent to make the best (optimum in some sense) decision.***

(NSMIA at p. 2).

Administrators do not have to concern about all the security problems. They interact with the agent from a high level using security policies. Security policies tell the agents what behavior they should exhibit when attacks occur. Hence, communications between agents permit to collect information. This information ***permits the agents to identify attacks*** that can not be detected if it is static. Giving more autonomy to the agent permits the system to react in “real time” to attacks and ***to take necessary actions*** to avoid severe consequences of the attack.

(NSMIA at p. 8).

2259. “We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. These entities become more intelligent, ***capable of making various decisions*** with autonomy to detect intrusions and to overcome their bad effects.” (NSMIA at p. 2)

18. ***Claim 29***

- a. ***The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.***

2260. NSMIA discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

To model intrusion detection, agents need to combine ***cognitive abilities*** (knowledge-based) to reason about complex situations, and reactive

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

abilities (stimulus-response). So, an agent may have two kinds of behaviors: *reactive* and *cognitive* behaviors.

(NSMIA at p. 12).

**Cognitive agents:** A cognitive agent *is able to find a solution for a complex problem* while communicating with other agents and interacting with its knowledge base. Its main features include a high reasoning capacity, data processing, perception, learning, control, communication and expertise per activity domain.

(NSMIA at p. 7).

Combined knowledge and experience of the agent with the information coming from neighboring agents permits *the agent to make the best (optimum in some sense) decision*. In this paper, we suggest to improve network security management by integrating DAI approach based on multi-agent system technique in Intrusion Detection Systems (IDS). We propose a new approach based on providing the NSM (Network Security Management) hosts with additional functionalities. *These entities become more intelligent, capable of making various decisions* with autonomy to detect intrusions and to overcome their bad effects.

(NSMIA at p. 2).

Cognitive agents: A cognitive agent is able to find a solution for a complex problem while communicating with other agents and interacting with its knowledge base. Its main features include a high reasoning capacity, data processing, *perception*, learning, control, communication and expertise per activity domain.

(NSMIA at p. 7).

19. **Claim 30**

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2261. NSMIA discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

*A deliberation module:* that enables agent intelligence and autonomy. The hybrid agent should be able to reason and extrapolate by relying on **built knowledge** and experience in a rational way. Decisions of the agent depend on the security environment status, the neighboring system evolution and its mental attitudes.

(NSMIA at p. 11).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The goal of IDS is to detect attacks especially in real-time fashion. These systems use one or both approaches of intrusion detection:

the behavior-based intrusion detection approach, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile;

the *knowledge-based intrusion detection* approach, which detects intrusions upon a comparison between parameters of the user's session and known *pattern attacks stored in a database*.

(NSMIA at p. 4).

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2262. NSMIA discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2263. NSMIA discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2264. NSMIA discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2265. NSMIA discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2266. NSMIA discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

2267. NSMIA discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2268. NSMIA discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

2269. NSMIA discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2270. NSMIA discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2271. NSMIA discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*assigned goal of the agent is expressed as a policy.*

2272. NSMIA discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2273. NSMIA discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2274. NSMIA discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2275. NSMIA discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2276. NSMIA discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2277. NSMIA discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2278. NSMIA discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

2279. As explained in detail below and in the chart attached as Ex. A-19, NSMIA anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

**O. Anticipation by and/or Obviousness in View of *INCA: An Agent-based Network Control Architecture*, published in the 1998 International Workshop on Intelligent Agents for Telecommunication Applications.**

2280. As explained in detail below and in the chart attached as Ex. A-13, INCA anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2281. INCA discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

This paper describes the design and implementation of INCA, **an open architecture for the distributed management of multi-service networks**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**and systems applications.** The Intelligent Network Control Architecture is populated by stationary and mobile intelligent agents. These agents perform monitoring and control of network and systems components, thereby supporting the integrated management of networks and services.

INCA at Abstract.

2282. Further examples include INCA at Section 5. *See generally, e.g.,* INCA at Section 4.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2283. INCA discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

This paper describes the design and implementation of INCA, an open architecture for the distributed management of multi-service networks and systems applications. The *Intelligent Network Control Architecture* is populated by stationary and mobile intelligent agents. ***These agents perform monitoring and control of network and systems components,*** thereby supporting the integrated management of networks and services.

INCA at Abstract.

2284. Further examples include INCA at Sections 4.1.1. , 4.1.2. *See generally, e.g.,* INCA at Section 4.

- c. *is able to communicate with other software agents in the computer network;*

2285. INCA discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

***Inter-agent communication. Communication between agents is offered by a common interface for inter-agent message invocation.*** This functionality includes a naming service from which agents can get references to other agents.

INCA at Section 4.1.2.

2286. Further examples include INCA at Section 4.

- d. *is capable of perceiving its own state*

2287. INCA discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- *Network Control Agents* support the network manager e.g. in configuration and fault management of the network. The tasks they pursue are demand driven, perhaps triggered by network faults, and often have to be carried out as fast as possible.
- *Service Management Agents* support the service provider, e.g. by a service subscription agent. They use the network environment together with the platform services to install and establish end user services, or reconfigure them.
- *Network Maintenance Agents* perform repeatedly occurring management tasks, e.g. the gathering of data from selected network elements or fault analysis. Their tasks are carried out repeatedly, only rarely requiring adjustments by the network management instance.
- *Network Monitoring Agents* perform routine tasks such as the filtering of raw data collected from network elements. Often, such agents are stationary, since their task is to monitor a concrete element of the network, or a link.”

INCA at Section 4.2.

2288. Further examples include INCA at Section 4.4. *See generally, e.g.,* INCA at Section 4.

e. *and is able to clone itself,*

2289. INCA discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

The architecture provides transaction capabilities to control transport and mobility of agents, agent prioritization, and multiple agent code transfer schemes. ***Managed objects used to access resources on network elements and new system functionalities can be created, distributed, and replaced dynamically.*** An example INCA application demonstrates that prioritized agents are necessary to support the timely execution of critical tasks.

INCA at Abstract.

2290. Further examples include INCA at Sections 4.1.1, 4.1.2, 4.4, 4.5. *See generally, e.g.,* INCA at Section 4.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2291. INCA discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**Repository. *Dedicated stations offer a repository service. This service provides access to the code off all agents that can run on the platform.***  
The repository can be accessed via the control service to push agent code to a station or by any station to pull agent code.

INCA at Section 4.1.2.

2292. Further examples include INCA at Sections 4.2, 7. *See generally, e.g.,* INCA at Section 4.

g. *monitoring the computer network;*

2293. INCA discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

- *Network Control Agents* support the network manager e.g. in configuration and fault management of the network. The tasks they pursue are demand driven, perhaps triggered by network faults, and often have to be carried out as fast as possible.
- *Service Management Agents* support the service provider, e.g. by a service subscription agent. They use the network environment together with the platform services to install and establish end user services, or reconfigure them.
- *Network Maintenance Agents* perform repeatedly occurring management tasks, e.g. ***the gathering of data from selected network elements or fault analysis***. Their tasks are carried out repeatedly, only rarely requiring adjustments by the network management instance.
- *Network Monitoring Agents* perform routine tasks such as the filtering of raw data collected from network elements. Often, such agents are stationary, ***since their task is to monitor a concrete element of the network, or a link.***

INCA at Section 4.2.

2294. Further examples include INCA at Section 4.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2295. INCA discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In order to demonstrate the usefulness of agent prioritization for network management applications, we chose a simple scenario. It consists of a set of connected LANs. At each LAN there is one network element hosting an INCA station. Each INCA station provides a set of managed objects allowing agents to access the network elements in the LAN. Furthermore, we consider a management station launching agents for monitoring and control. Monitoring agents are stationary and monitor dedicated network elements of a LAN. Thus, there typically are more than one monitoring agents at a station. Different to these continuously running monitoring agents, there are control agents which are launched interactively by a network operator to perform a usually short task. For these agents instantaneous task completion is desired.

INCA at Section 5.

2296. Further examples include INCA at Sections 3, 4.2, 7. *See generally, e.g.,* INCA at Section 4.

- i. *including predicting a failure of a network component based on a prediction algorithm*

2297. INCA discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

So far, INCA agents are implemented in a straightforward manner, with the algorithms coded directly into them. Although this approach is sufficient for smaller applications and quick prototyping, ***we are going to extend the INCA platform by libraries for agent intelligence.*** Hence, our main work is currently focused on choosing a proper goal representation and problem solving formalism.

INCA at Section 7

2298. Further examples include INCA at Sections 3, 4.2, 5. *See generally, e.g.,* INCA at Section 4.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2299. INCA discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

In order to demonstrate the usefulness of agent prioritization for network management applications, we chose a simple scenario. It consists of a set of connected LANs. At each LAN there is one network element hosting an INCA station. Each INCA station provides a set of managed objects allowing agents to access the network elements in the LAN. Furthermore, we consider a management station launching agents for monitoring and control. Monitoring agents are stationary and monitor dedicated network elements of a LAN. Thus, there typically are more than one monitoring

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agents at a station. Different to these continuously running monitoring agents, there are control agents which are launched interactively by a network operator to perform a usually short task. For these agents instantaneous task completion is desired.

INCA at Section 5.

2300. Further examples include INCA at Sections 3, 4.2, 7. *See generally, e.g.,* INCA at Section 4.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2301. INCA discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl.

1. For example:

**Loading of agent code. Agent code can be transferred between stations.**

Three schemes for code transfer are supported: push, pull, and migrate. Pushed code is sent from an agent code repository to a dedicated station, pulled code is loaded by the station from a repository, and in migrating code is sent from one (non-repository) station to another. All stations support the same code format, i.e. the same code can be executed at any stations. In order to improve performance, caching of agent code is used.

**Transfer of agent state.** The current state of an agent can be transferred from one station to another. In combination with agent code transfer, this service enables agent migration.

INCA at Section 4.1.1.

2302. Further examples include INCA at Sections 4.1.2, 4.5, 7. INCA at Fig. 2. *See generally, e.g.,* INCA at Section 4.

2303. To the extent that this limitation is not expressly disclosed by INCA, this Claim Limitation was obvious to a person of ordinary skill in the art because an agent dynamically choosing the next station to migrate to implies using a predictive algorithm or model to determine an optimal policy.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

2304. INCA discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

**Inter-agent communication. Communication between agents is offered by a common interface for inter-agent message invocation.** This functionality includes a naming service from which agents can get references to other agents.

INCA at Section 4.1.2.

2305. Further examples include INCA at Section 4.5; INCA at Fig. 2. *See generally, e.g.,* INCA at Sections 3, 4.

2306. To the extent this limitation is not explicitly disclosed by INCA, it would have been obvious to a person of ordinary skill in the art. INCA contemplates software agents relying on code and functions from a repository, and discloses agents exchanging messages between one another. It would have been obvious to a person of ordinary skill in the art at the time of invention to modify INCA so that an agent may request a policy from another entity (agent, repository, etc.) when it requires additional functionality, or when another agent reports superior results.

2. **Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2307. INCA discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

**Repository. Dedicated stations offer a repository service. This service provides access to the code off all agents that can run on the platform.** The repository can be accessed via the control service to push agent code to a station or by any station to pull agent code.

INCA at Section 4.1.2.

2308. Further examples include INCA at Sections 4.2, 7. *See generally, e.g.,* INCA at Section 4.

3. **Claim 3**

- a. *The method of claim 1, further comprising: obtaining information*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*about a network component by the software agent in performing the predefined task; and*

2309. INCA discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

- *Network Control Agents* support the network manager e.g. in configuration and fault management of the network. The tasks they pursue are demand driven, perhaps triggered by network faults, and often have to be carried out as fast as possible.
- *Service Management Agents* support the service provider, e.g. by a service subscription agent. They use the network environment together with the platform services to install and establish end user services, or reconfigure them.
- *Network Maintenance Agents* perform repeatedly occurring management tasks, e.g. ***the gathering of data from selected network elements or fault analysis***. Their tasks are carried out repeatedly, only rarely requiring adjustments by the network management instance.
- *Network Monitoring Agents* perform routine tasks such as the filtering of raw data collected from network elements. Often, such agents are stationary, ***since their task is to monitor a concrete element of the network, or a link.***”

INCA at Section 4.2.

2310. Further examples include INCA at Section 4.

- b. *constructing a topological representation of the computer network from the information.*

2311. INCA discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

***The architecture provides transaction capabilities to control transport and mobility of agents***, agent prioritization, and multiple agent code transfer schemes. Managed objects used to access resources on network elements and new system functionalities can be created, distributed, and replaced dynamically. An example INCA application demonstrates that prioritized agents are necessary to support the timely execution of critical tasks.

INCA at Abstract.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2312. Further examples include INCA at Sections 4.1.2, 4.4. *See generally, e.g.*, INCA at Section 4.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2313. INCA discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

**Itinerary control.** The control service allows to link a mobile agent with a predefined itinerary. The itinerary becomes part of the agent state, but is not accessed by the agent itself. ***It is controlled by a station whenever a mobile agent is to be migrated, in order to determine the station to send it to.*** When an agent with a predefined itinerary is launched by the control agent, a copy of the agents itinerary is sent to the locator agent which compares the itinerary with location messages.

INCA at Section 4.1.2.

2314. Further examples include INCA at Sections 3, 5, 7. *See generally, e.g.*, INCA at Section 4.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2315. INCA discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

**Itinerary control.** The control service allows to link a mobile agent with a predefined itinerary. The itinerary becomes part of the agent state, but is not accessed by the agent itself. ***It is controlled by a station whenever a mobile agent is to be migrated, in order to determine the station to send it to.*** When an agent with a predefined itinerary is launched by the control agent, a copy of the agents itinerary is sent to the locator agent which compares the itinerary with location messages.

INCA at Section 4.1.2.

2316. Further examples include INCA at Sections 3, 4.2, 5, 7. *See generally, e.g.*, INCA at Section 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6. *Claim 7*

a. *A computer network, comprising:*

2317. INCA discloses the preamble of claim 7, “[a] computer network, comprising.” ’730

Pat. at Cl. 7. For example:

This paper describes the design and implementation of *INCA, an open architecture for the distributed management of multi-service networks and systems applications*. The *Intelligent Network Control Architecture* is populated by stationary and mobile intelligent agents. These agents perform monitoring and control of network and systems components, thereby supporting the integrated management of networks and services.

INCA at Abstract.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2318. INCA discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

2319. INCA discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

2320. INCA discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2321. INCA discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2322. INCA discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2323. INCA discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

**Loading of agent code. Agent code can be transferred between stations.**

Three schemes for code transfer are supported: push, pull, and migrate. Pushed code is sent from an agent code repository to a dedicated station, pulled code is loaded by the station from a repository, and in migrating code is sent from one (non-repository) station to another. All stations support the same code format, i.e. the same code can be executed at any stations. In order to improve performance, caching of agent code is used.

INCA at Section 4.1.1.

2324. Further examples include INCA at Sections 4.1, 4.1.1., 4.1.2, 4.5. *See generally*, e.g., INCA at Section 4.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2325. INCA discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2326. INCA discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2327. INCA discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the optimal policy;*

2328. INCA discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2329. INCA discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2330. INCA discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2331. INCA discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

***Inter-agent communication. Communication between agents is offered by a common interface for inter-agent message invocation.*** This functionality includes a naming service from which agents can get references to other agents.

INCA at Section 4.1.2.

2332. Further examples include INCA at Section 4.3. *See generally, e.g.,* INCA at Section 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2333. INCA discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

*Also we are aware of the importance of security in agent systems, in particular in network management. Various security mechanisms have already been proposed, but it is still subject to research which of them is suited best for mobile agents in network management.*

INCA at Section 7.

2334. Further examples include INCA at Sections 4, 7.

2335. To the extent this limitation is not explicitly disclosed by INCA, it would have been obvious to a person of ordinary skill in the art. INCA discloses using security mechanisms in a network environment. It would have been obvious to a person of ordinary skill in the art at the time of invention to modify INCA by using a secure communications protocol.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2336. INCA discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

**4.1.2 Global Services**

Besides services provided by each single station there are services offered by the entire platform, or by a set of dedicated stations:

**Control.** For each INCA platform there is at least one *control station*. This station executes a stationary control agent offering a control service for launching and monitoring agents. *Usually, the location of this station is the console of the network manager.*

**Repository.** Dedicated stations offer a repository service. This service provides access to the code off all agents that can run on the platform. The repository can be accessed via the control service to push agent code to a station or by any station to pull agent code.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**Locator.** A unique station runs a stationary locator agent offering the location tracking service. The locator agent receives messages about monitored agents from the stations. Based on this information the locator agent answers requests from the control agent about location and status of other agents. Furthermore, the locator agent supervises predefined itineraries (see below).

**Itinerary control.** The control service allows to link a mobile agent with a predefined itinerary. The itinerary becomes part of the agent state, but is not accessed by the agent itself. It is controlled by a station whenever a mobile agent is to be migrated, in order to determine the station to send it to. When an agent with a predefined itinerary is launched by the control agent, a copy of the agents itinerary is sent to the locator agent which compares the itinerary with location messages.

**Migration control.** By combining itinerary control with persistence of agent state fault tolerance can be increased for the migration of agents.

**Inter-agent communication.** Communication between agents is offered by a common interface for inter-agent message invocation. This functionality includes a naming service from which agents can get references to other agents.

INCA at Section 4.1.2.

2337. Further examples include INCA at Section 4.1.1. *See generally, e.g.,* INCA at Section 4.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2338. INCA discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.”

’730 Pat. at Cl. 17. For example:

**4.1.2 Global Services**

Besides services provided by each single station there are services offered by the entire platform, or by a set of dedicated stations:

**Control.** For each INCA platform there is at least one *control station*. This station executes a stationary control agent offering a control service for launching and monitoring agents. *Usually, the location of this station is the console of the network manager.*

**Repository.** Dedicated stations offer a repository service. This service provides access to the code off all agents that can run on the platform. The

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

repository can be accessed via the control service to push agent code to a station or by any station to pull agent code.

**Locator.** A unique station runs a stationary locator agent offering the location tracking service. The locator agent receives messages about monitored agents from the stations. Based on this information the locator agent answers requests from the control agent about location and status of other agents. Furthermore, the locator agent supervises predefined itineraries (see below).

**Itinerary control.** The control service allows to link a mobile agent with a predefined itinerary. The itinerary becomes part of the agent state, but is not accessed by the agent itself. It is controlled by a station whenever a mobile agent is to be migrated, in order to determine the station to send it to. When an agent with a predefined itinerary is launched by the control agent, a copy of the agents itinerary is sent to the locator agent which compares the itinerary with location messages.

**Migration control.** By combining itinerary control with persistence of agent state fault tolerance can be increased for the migration of agents.

**Inter-agent communication.** Communication between agents is offered by a common interface for inter-agent message invocation. This functionality includes a naming service from which agents can get references to other agents”

INCA at Section 4.1.2.

2339. Further examples include INCA at Section 4.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2340. INCA discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

**Support for location and status monitoring of agents.** Location and status of an agent migrating from station to station may be monitored by a central instance called *locator*. For monitored agents the station send messages to the locator on arrival and departure *or termination of the agent*.

INCA at Section 4.1.1.

2341. Further examples include INCA at Sections 4.2, 4.4. *See generally, e.g.,* INCA at Section 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2342. INCA discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

**Repository.** Dedicated stations offer a repository service. *This service provides access to the code off all agents that can run on the platform.* The repository can be accessed via the control service to push agent code to a station or by any station to pull agent code.

INCA at Section 4.1.2.

2343. Further examples include INCA at Section 4.1.2. *See generally, e.g.,* INCA at Section 4.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2344. INCA discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

**4.1.2 Global Services**

Besides services provided by each single station there are services offered by the entire platform, or by a set of dedicated stations:

**Control.** For each INCA platform there is at least one *control station*. This station executes a stationary control agent offering a control service for launching and monitoring agents. *Usually, the location of this station is the console of the network manager.*

**Repository.** Dedicated stations offer a repository service. This service provides access to the code off all agents that can run on the platform. The repository can be accessed via the control service to push agent code to a station or by any station to pull agent code.

**Locator.** A unique station runs a stationary locator agent offering the location tracking service. The locator agent receives messages about monitored agents from the stations. Based on this information the locator agent answers requests from the control agent about location and status of other agents. Furthermore, the locator agent supervises predefined itineraries (see below).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**Itinerary control.** The control service allows to link a mobile agent with a predefined itinerary. The itinerary becomes part of the agent state, but is not accessed by the agent itself. It is controlled by a station whenever a mobile agent is to be migrated, in order to determine the station to send it to. When an agent with a predefined itinerary is launched by the control agent, a copy of the agents itinerary is sent to the locator agent which compares the itinerary with location messages.

**Migration control.** By combining itinerary control with persistence of agent state fault tolerance can be increased for the migration of agents.

**Inter-agent communication.** Communication between agents is offered by a common interface for inter-agent message invocation. This functionality includes a naming service from which agents can get references to other agents.

INCA at Section 4.1.2.

2345. Further examples include INCA at Section 4.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2346. INCA discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

To ensure effective and efficient function of large networks as well as provision of services, and furthermore to address the requirements of network providers, service providers, retailers and users, there is a strong need for a coherent framework supporting automated and more intelligent end-to-end management solutions.

INCA at Section 3.

2347. Further examples include INCA at Section 4.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2348. INCA discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

2349. INCA discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl.

26. For example:

The architecture provides transaction capabilities to control transport and mobility of agents, agent prioritization, and multiple agent code transfer schemes. Managed objects used to access resources on network elements and ***new system functionalities can be created, distributed, and replaced dynamically.*** An example INCA application demonstrates that prioritized agents are necessary to support the timely execution of critical tasks.

INCA at Abstract.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2350. INCA discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

To ensure effective and efficient function of large networks as well as provision of services, and furthermore to address the requirements of network providers, service providers, retailers and users, there is a strong need for a coherent framework supporting automated and more intelligent end-to-end management solutions.

Network management today is typically based on a client-server model with SNMP [2] and CMIP [16] as the de facto standards for monitoring and for managing a network of computers and devices. Managed objects, like hosts, routers, bridges, switches, printers etc., provide read/write access to a set of variables through a management process. A management station can then communicate with the management processes in a client-server relationship.

Although simple, this centralized approach to network management has some severe drawbacks with regard to scalability, flexibility and performance. By dividing management functions into mobile, autonomous and intelligent computing entities (i.e. software agents), many of the problems with network and service management can be addressed:

- Scalability is increased, as management is not performed solely by the management station but delegated to distributed management agents.
- ***Repetitive tasks can be avoided if software agents learn from experience.***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- *Fault tolerance can be increased through the autonomy and learning capability of management agents.*
- Better performance is achieved by moving the management functionality closer to the actual network element, thus reducing network traffic.
- The low-level details of different devices can be hidden behind the agent interface.
- Legacy systems can be integrated by using an agent for inter-operation.

Typical scenarios like service provision through different administrative domains, including accounting and charging, can be handled by a multi-agent based approach to distributed network and service management in a more scalable and coherent way than with a centralized approach to network management.

This is particularly true for the creation of new services in the telecommunications world. With the advent of high-speed networks based on ATM switching, there is demand for better support when creating and managing new services. Proprietary, low level interfaces to network devices currently prevent a rapid development of telecommunication applications (see [17]). Control and management of scalable multi service networks is difficult since the switching software is usually tightly coupled with the individual switching devices.

INCA at Section 3.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2351. INCA discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

Network management today is typically based on a client-server model with SNMP [2] and CMIP [16] as the de facto ***standards for monitoring and for managing a network of computers and devices***. Managed objects, like hosts, routers, bridges, switches, printers etc., provide read/write access to a set of variables through a management process. A management station can then communicate with the management processes in a client-server relationship.

INCA at Section 3.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2352. Further examples include INCA at Sections 4.1, 4.1.1, 4.1.2, 5. *See generally, e.g.,* INCA at Section 4.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2353. INCA discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2354. INCA discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2355. INCA discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2356. INCA discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2357. INCA discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2358. INCA discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

- h. *creating test policy and modeling a behavior of the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network based on the test policy to determine an optimal policy for the computer network*

2359. INCA discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

2360. INCA discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2361. INCA discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2362. INCA discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2363. INCA discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2364. INCA discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

b. *constructing a topological representation of the computer network from the information.*

2365. INCA discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2366. INCA discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2367. INCA discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

a. *The machine-readable storage medium of claim 33, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*determining uses a numerical method.*

2368. INCA discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2369. INCA discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

2370. I expect to testify that INCA anticipates and/or renders obvious each Asserted Claim of the '730 Patent. A claim chart illustrating that INCA discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-13.

**P. Anticipation by and/or Obviousness in View of *Distributed Network Security Management Using Intelligent Agents*, by Boudaoud, et al., published in April 1998 at the HP Openview University Association Workshop.**

2371. As explained in detail below and in the chart attached as Ex. A-14, Boudaoud anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2372. Boudaoud discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” '730 Pat. at Cl. 1. For example:

Intrusion detection is a practical approach ***for enhancing the security of computer and network systems***. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database. The behavior-

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. ***We focus our work on network intrusion detection systems and we present below two specific systems DIDS (Distributed Intrusion Detection System) and CSM (Co-operating Security Managers).***

Boudaoud at Sec. 1.

2373. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 1, 3.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2374. Boudaoud discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

***User Interface Skill: this skill permit the security officer to transmit to the agents, particularly to the MA, some requests, like a new security policy or a new intrusion to detect or a new security event to monitor, ...It sends the security officer requests to the Security Manager Skill.***

Boudaoud at Sec. 5.3.

2375. Further examples include Boudaoud at Secs. 2, 5.1, 5.4; Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

- c. *is able to communicate with other software agents in the computer network;*

2376. Boudaoud discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

***Both the Communication Module, which is responsible for managing interactions with the other agents and the Social Manager, which holds information about the other agents, support inter-agent communication facilities in the agent.***

Boudaoud at Sec. 3.1.

2377. Further examples include Boudaoud at Secs. 2, 5.4. *See generally, e.g.,* Boudaoud at Sections 3, 5.

- d. *is capable of perceiving its own state*

2378. Boudaoud discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Nevertheless, different types of agents reflect a set of properties, which common among them and are described below [12]:

- Autonomy: is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal and/or external environments
- Co-operation: an Agent is co-operative and is able to have a social ability. This sociability allows an agent to interact with other agents for the purpose of performing tasks that are beyond the capability of a particular agent. This capability goes from delegation (distribution of sub-tasks) to peer-to-peer interworking.
- Proactiveness: it is the agent's ability to anticipate situations and change its course of action to avoid them. Proactive agents are capable of exhibiting goal-direct behaviors by taking some initiative [13][14].
- Reactivity: this kind of behavior means that the agent reacts in real-time to changes that occur in its environments.
- Adaptability: is the ability of an agent to modify its behavior over time to fulfill its problem-solving goal.
- Intelligence: the term "Intelligence" means that the agent is able to exhibit a certain level of intelligence priority, ranging from predefined actions (planning) up to self-learning (define new actions).
- Flexibility: is the ability an agent should have to adapt itself to cope with the environment in which it is situated.
- Mobility: an Agent is mobile. It is capable of moving from one localisation to another in order to perform a particular task or to react to a particular event

Having studied the properties of the [Intelligent Agent (IA)] and the aspects and requirements of a security management, it can be concluded that IA provides a more coherent and flexible approach of security management. The security management architecture based on the concept of IA can be conceived as if it were made of the autonomous

IAs co-operating with each other to achieve Global Security Policy. The section 5 describes the security management architecture.

Boudaoud at Sec. 2.

2379. Further examples include Boudaoud at Sec. 5.4. *See generally, e.g.,* Boudaoud at Sections 3, 5.

e. *and is able to clone itself,*

2380. Boudaoud discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Nevertheless, different types of agents reflect a set of properties, which common among them and are described below [12]:

- **Autonomy:** is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal and/or external environments
- **Co-operation:** an Agent is co-operative and is able to have a social ability. This sociability allows an agent to interact with other agents for the purpose of performing tasks that are beyond the capability of a particular agent. This capability goes from delegation (distribution of sub-tasks) to peer-to-peer interworking.
- **Proactiveness:** it is the agent's ability to anticipate situations and change its course of action to avoid them. Proactive agents are capable of exhibiting goal-direct behaviors by taking some initiative [13][14].
- **Reactivity:** this kind of behavior means that the agent reacts in real-time to changes that occur in its environments.
- **Adaptability:** is the ability of an agent to modify its behavior over time to fulfill its problem-solving goal.
- **Intelligence:** the term "Intelligence" means that the agent is able to exhibit a certain level of intelligence priority, ranging from predefined actions (planning) up to self-learning (define new actions).
- **Flexibility:** is the ability an agent should have to adapt itself to cope with the environment in which it is situated.
- **Mobility:** an Agent is mobile. It is capable of moving from one localisation to another in order to perform a particular task or to react to a particular event

Having studied the properties of the [Intelligent Agent (IA)] and the aspects and requirements of a security management, it can be concluded that IA provides a more coherent and flexible approach of security management. The security management architecture based on the concept of IA can be conceived as if it were made of the autonomous

IAs co-operating with each other to achieve Global Security Policy. The section 5 describes the security management architecture.”

Boudaoud at Sec. 2.

2381. As an additional example, the agents can clone themselves because they can copy part of their skill base to another agent.

**Skills** can be downloaded dynamically into the agent inside its *Skill Base*. *The main role of the Skill Manager is to check the availability of pre-requisite skills required by newly loaded skills and if they are not yet loaded, it must search for them either locally or on distant agents.* It is also responsible for disposing off no useful skills to keep the agent's size as



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

small as possible. During its operation, the skill can update or delete existing beliefs or create new ones. A skill operation may depend on beliefs created by other skills, and the *Skill Manager* is therefore in charge of dispatching asynchronously these beliefs to the interested skill in a transparent way. It holds all the necessary information about the skills in the *Skill Base*.

Boudaoud at Sec. 3.1.

2382. Further examples include Boudaoud at Sec. 5.2; Boudaoud at Fig. 1. *See generally*, e.g., Boudaoud at Sections 3, 5.

2383. To the extent that this limitation is not expressly disclosed by Boudaoud, this limitation would have been obvious to a person of ordinary skill in the art because intelligent and autonomous agents would clone themselves if the situation required it. This limitation was further obvious, because intelligent and autonomous agents with skill bases would clone parts of their skill bases into other agents if the situation required it.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2384. Boudaoud discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

**User Interface Skill:** *this skill permit the security officer to transmit to the agents, particularly to the MA, some requests, like a new security policy or a new intrusion to detect or a new security event to monitor, ...It sends the security officer requests to the Security Manager Skill.*

Boudaoud at Sec. 5.3.

2385. Further examples include Boudaoud at Secs. 2, 5.4. *See generally*, e.g., Boudaoud at Sections 3, 5.

g. *monitoring the computer network;*

2386. Boudaoud discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1.

The **Security Management Agent (SMA)** is an intelligent agent that collects, filters management information and performs security management activities. The management activities are defined by the administrator and reflect the Security Policy. Thus, network environment is populated by a set of SMA that co-operate with each other in order to perform global security management activities (described in the following



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure). We have identified two SMA types: Master SMA (MSMA) and Slave SMA (SSMA). The SSMA is responsible for managing the security of his domain constituted of several hosts. There are several SSMA that performs some analysis before informing the MSMA when they suspect an attack. The MSMA is responsible for coordinating SSMA tasks and correlating information received from SSMA. The MSMA, in his turn makes his own analysis to confirm or detect an occurred attack and take appropriate actions (like informing the security officer (S.O)). The SSMA can communicate and co-operate before sending their reports to the MSMA. ***We identified a specific agent named External Agent, which its role is to manage all activities going in or out the monitored network.***

Boudaoud at Sec. 5.1.

2387. Further examples include Boudaoud at Boudaoud at Sec. 5.2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2388. Boudaoud discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

In this section, we present the DIANA1 agent architecture [15]. The DIANA Architecture is the shell of our system. The key characteristics of these agents are their ability to acquire new capabilities and skills, without interrupting their operations, permit network management applications to be easily adaptable when changes occur. The DIANA agent architecture consists of two main component types: the Brain, which is responsible for managing agent skills and the skills, which provide the agent with capabilities and behaviors.

Boudaoud at Sec. 3.

2389. Further examples include Boudaoud at Secs. 3, 5.1; Boudaoud at Fig. 1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

- i. *including predicting a failure of a network component based on a prediction algorithm*

2390. Boudaoud discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

In this section, we present the DIANA1 agent architecture [15]. The DIANA Architecture is the shell of our system. The key characteristics of these agents are their ability to acquire new capabilities and skills, without interrupting their operations, permit network management applications to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

be easily adaptable when changes occur. The DIANA agent architecture consists of two main component types: the Brain, which is responsible for managing agent skills and the skills, which provide the agent with capabilities and behaviors.

Boudaoud at Sec. 3.

2391. Further examples include Boudaoud at Secs. 3, 5.1; Boudaoud at Fig. 1. *See generally, e.g.*, Boudaoud at Sections 3, 5.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2392. Boudaoud discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

In this section, we present the DIANA1 agent architecture [15]. The DIANA Architecture is the shell of our system. The key characteristics of these agents are their ability to acquire new capabilities and skills, without interrupting their operations, permit network management applications to be easily adaptable when changes occur. The DIANA agent architecture consists of two main component types: the Brain, which is responsible for managing agent skills and the skills, which provide the agent with capabilities and behaviors.

Boudaoud at Sec. 3.

2393. Further examples include Boudaoud at Secs. 3, 5.1; Boudaoud at Fig. 1. *See generally, e.g.*, Boudaoud at Sections 3, 5.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2394. Boudaoud discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

In this section, we present the DIANA1 agent architecture [15]. The DIANA Architecture is the shell of our system. The key characteristics of these agents are their ability to acquire new capabilities and skills, without interrupting their operations, permit network management applications to be easily adaptable when changes occur. The DIANA agent architecture consists of two main component types: the Brain, which is responsible for managing agent skills and the skills, which provide the agent with capabilities and behaviors.

Boudaoud at Sec. 3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2395. Further examples include Boudaoud at Secs. 3, 5.1, 5.2, 5.4; Boudaoud at Fig. 1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2396. Boudaoud discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

**Skills** can be downloaded dynamically into the agent inside its Skill Base. *The main role of the Skill Manager is to check the availability of pre-requisite skills required by newly loaded skills and if they are not yet loaded, it must search for them either locally or on distant agents.*

Boudaoud at Sec. 3.1.

2397. Further examples include Boudaoud at Secs. 2, 5.2; Boudaoud at Fig. 1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

2. **Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2398. Boudaoud discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

**User Interface Skill:** *this skill permit the security officer to transmit to the agents, particularly to the MA, some requests, like a new security policy or a new intrusion to detect or a new security event to monitor, ...It sends the security officer requests to the Security Manager Skill.*

Boudaoud at Sec. 5.3.

2399. Further examples include Boudaoud at Secs. 2, 5.4. *See generally, e.g.,* Boudaoud at Sections 3, 5.

3. **Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the predefined task; and*

2400. Boudaoud discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

The **Security Management Agent (SMA)** is an intelligent agent that collects, filters management information and performs security management activities. The management activities are defined by the administrator and reflect the Security Policy. Thus, network environment is populated by a set of SMA that co-operate with each other in order to perform global security management activities (described in the following Figure). We have identified two SMA types: Master SMA (MSMA) and Slave SMA (SSMA). The SSMA is responsible for managing the security of his domain constituted of several hosts. There are several SSMA that performs some analysis before informing the MSMA when they suspect an attack. The MSMA is responsible for coordinating SSMA tasks and correlating information received from SSMA. The MSMA, in his turn makes his own analysis to confirm or detect an occurred attack and take appropriate actions (like informing the security officer (S.O)). The SSMA can communicate and co-operate before sending their reports to the MSMA. ***We identified a specific agent named External Agent, which its role is to manage all activities going in or out the monitored network.***

Boudaoud at Sec. 5.1.

2401. Further examples include Boudaoud at Boudaoud at Sec. 5.2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

b. *constructing a topological representation of the computer network from the information.*

2402. Boudaoud discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

[Distributed Intrusion Detection System (DIDS)] operates on a local area network (LAN) and its architecture combines distributed monitoring and data reduction with centralized data analysis [3]. A DIDS director, a LAN monitor, and a series of host monitor constitute it. The LAN monitor reports to the DIDS director unauthorized or suspicious activities on the network. ***The host monitors collect audit data for the individual host and perform some simple analysis on the data.*** The relevant information is then transmitted to the DIDS director. ***This director is responsible for analyzing all these data and detecting possible attacks.*** A shortcoming of DIDS is that the centralized nature of DIDS will limit its usefulness in wide area networks where communication with a central director from all hosts may swamp portions of the network.

Boudaoud at Sec. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2403. Further examples include Boudaoud at Sec. 2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

2404. To the extent that this limitation is not expressly disclosed by Boudaoud, this Claim Limitation was obvious to a person of ordinary skill in the art because constructing topological representations was a well-known method for analyzing network data at the time.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2405. Boudaoud discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

[Distributed Intrusion Detection System (DIDS)] operates on a local area network (LAN) and its architecture combines distributed monitoring and data reduction with centralized data analysis [3]. A DIDS director, a LAN monitor, and a series of host monitor constitute it. The LAN monitor reports to the DIDS director unauthorized or suspicious activities on the network. ***The host monitors collect audit data for the individual host and perform some simple analysis on the data.*** The relevant information is then transmitted to the DIDS director. ***This director is responsible for analyzing all these data and detecting possible attacks.*** A shortcoming of DIDS is that the centralized nature of DIDS will limit its usefulness in wide area networks where communication with a central director from all hosts may swamp portions of the network.

Boudaoud at Sec. 1.

2406. Further examples include Boudaoud at Sections 3, 5.

2407. To the extent that this limitation is not expressly disclosed by Boudaoud, this Claim Limitation was obvious to a person of ordinary skill in the art because numerical methods were a well-known method for analyzing network data at the time.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2408. Boudaoud discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

The [Intelligent Agents (IA)] should monitor the network in order to detect security-relevant events and then react according to the behaviour specified

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by the administrator. The IAs may also report the administrator Workstation the security-relevant events. In case of a special event, the IAs may also co-operate to check or have some information in order to have a more precise status on the special event. For example, if an agent detects an "unauthorizedAccessAttempt", it can co-operate with others agents to check if there are other login attempts on their hosts. An example of this functionality is given below. Suppose that an intruder came from an external network, in the night or in the weekend, obtained an access, and had an unauthorised activity. The agent that is monitoring all the incoming connections detects an "unknownAddress" and an "outOfHoursActivity" event. This agent can track the intruder by migrating to the host were the intruder is working. If the intruder "travel" from one host to another host, migrating agent can follows intruder's activities by co-operating with the others agents, responsible for monitoring these hosts. If one of the co-operating agents detects, for instance an "unauthorizedAccessAttempt", or an "suspiciousActivitiy", ***the first agent can migrate to the host on the entry of the internal network and close the connection or to ask another agent to do it.***

Boudaoud at Sec. 5.2.

2409. Further examples include Boudaoud at Secs. 2, 5.3, 5.4. *See generally, e.g.,* Boudaoud at Sections 3, 5.

2410. To the extent that this limitation is not expressly disclosed by Boudaoud, this Claim Limitation was obvious to a person of ordinary skill in the art because Dijkstra's Self Stabilizing Algorithm was a well-known algorithm for network management at the time.

6. ***Claim 7***

a. *A computer network, comprising:*

2411. Boudaoud discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. For example:

Intrusion detection is a practical approach ***for enhancing the security of computer and network systems.*** The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach, which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. ***We focus our work on network intrusion detection systems and we present below two specific systems DIDS***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

(Distributed Intrusion Detection System) and CSM (Co-operating Security Managers).

Boudaoud at Sec. 1.

2412. Further examples include Boudaoud at Fig. 2. *See generally, e.g.*, Boudaoud at Sections 1, 3.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2413. Boudaoud discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

2414. Boudaoud discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

2415. Boudaoud discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

2416. Boudaoud discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

- f. *is able to clone itself;*

2417. Boudaoud discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

- g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2418. Boudaoud discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In this section, we present the DIANA1 agent architecture [15]. ***The DIANA Architecture is the shell of our system.*** The key characteristics of these agents are their ability to acquire new capabilities and skills, without interrupting their operations, permit network management applications to be easily adaptable when changes occur. The DIANA agent architecture consists of two main component types: the Brain, which is responsible for managing agent skills and the skills, which provide the agent with capabilities and behaviors.

Boudaoud at Sec. 3.

2419. Further examples include Boudaoud at Sec. 1; Boudaoud at Fig. 1. *See generally*, e.g., Boudaoud at Sections 1, 3.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2420. Boudaoud discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2421. Boudaoud discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2422. Boudaoud discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2423. Boudaoud discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2424. Boudaoud discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2425. Boudaoud discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2426. Boudaoud discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

Both the ***Communication Module***, which is responsible for managing interactions with the other agents and the ***Social Manager***, which holds information about the other agents, support inter-agent communication facilities in the agent.

Boudaoud at Sec. 3.1.

2427. Further examples include Boudaoud at Sections 3, 5.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism comprises a secure communications protocol.*

2428. Boudaoud discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

Both the *Communication Module, which is responsible for managing interactions with the other agents* and the Social Manager, which holds information about the other agents, *support inter-agent communication facilities in the agent.*

Boudaoud at Sec. 3.1.

2429. Further examples include Boudaoud at Secs. 2, 5.4. *See generally, e.g.,* Boudaoud at Sections 3, 5.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2430. Boudaoud discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

The Management Agent Execution Environment (MAEE) is a set of components necessary for the execution and the migration of [intelligent agents].

Boudaoud at Sec. 5.1.

2431. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2432. Boudaoud discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The architecture relies on many [intelligent agents (IA)] for assuring intrusion detection. The IAs operate autonomously but according to a predefined security policy. *These policies can be defined at the initialisation of the IA or dynamically according to the global business policy*"

Boudaoud at Sec. 5.2.

2433. Further examples include Boudaoud at Sec. 5.2. *See generally, e.g.,* Boudaoud at Sections 3, 5.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2434. Boudaoud discloses claim 18, "[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend." '730 Pat. at Cl. 18. For example:

The Management Agent Factory (MAF) is an environment, in which security management *intelligent agents are created, initiated, resumed, and controlled*. The environment also serves as an access point for network security administrator.

Boudaoud at Sec. 5.1.

2435. Further examples include Boudaoud at Sec. 5.1. *See generally, e.g.,* Boudaoud at Sections 3, 5.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2436. Boudaoud discloses claim 19, "[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment." '730 Pat. at Cl. 19. For example:

Intrusion detection is a practical approach *for enhancing the security of computer and network systems*. The goal of IDS is to detect attacks especially in real-time fashion. There are systems based on host audit-trail and/or network traffic analysis to detect suspicious activity. These systems use one or both of two approaches of intrusion detection. The first approach is the behavior-based intrusion detection, which discovers intrusive activity by comparing the user or system behavior with a normal behavior profile. The second approach is a knowledge-based intrusion detection approach,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

which detects intrusions upon a comparison between parameters of the user's session and known pattern attacks stored in a database. The behavior-based intrusion detection approach allows detecting unknown intrusions contrarily to the knowledge-based intrusion detection approach, which detects well-known intrusions. *We focus our work on network intrusion detection systems and we present below two specific systems DIDS (Distributed Intrusion Detection System) and CSM (Co-operating Security Managers).*

Boudaoud at Sec. 1.

2437. Further examples include Boudaoud at Fig. 2. *See generally, e.g.,* Boudaoud at Sections 1, 3.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2438. Boudaoud discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

***Skills can be downloaded dynamically into the agent inside its Skill Base.***  
The main role of the Skill Manager is to check the availability of pre-requisite skills required by newly loaded skills and if they are not yet loaded, it must search for them either locally or on distant agents. It is also responsible for disposing off no useful skills to keep the agent's size as small as possible. During its operation, the skill can update or delete existing beliefs or create new ones. A skill operation may depend on beliefs created by other skills, and the *Skill Manager* is therefore in charge of dispatching asynchronously these beliefs to the interested skill in a transparent way. It holds all the necessary information about the skills in the *Skill Base*.

Boudaoud at Sec. 3.1.

2439. Further examples include Boudaoud at Sections 3, 5.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2440. Boudaoud discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

**[Co-operating Security Managers (CSM)]** was designed to perform intrusion detection in a distributed environment [4]. A CSM must be run on each computer connected to a network to facilitate the co-operative detection of network intrusions. It consists of following parts:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- a local intrusion detection component. It performs intrusion detection for the local host and is responsible for proactive detection of attacks on other host;
- a security manager which co-ordinates the distributed intrusion detection between CSMs;
- an intruder handling component. Its role is to take actions when an intruder is detected;
- *a graphical user interface;*
- a command monitor which intercepts the commands executed by a user and sends for analysis;
- a TCP communication module.”

Boudaoud at Sec. 1.

2441. Further examples include Boudaoud at Sections 3, 5.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2442. Boudaoud discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2443. Boudaoud discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

**Skills** can be downloaded dynamically into the agent inside its *Skill Base*. The main role of the Skill Manager is to check the availability of prerequisite skills required by newly loaded skills and if they are not yet loaded, it must search for them either locally or on distant agents. It is also responsible for disposing off no useful skills to keep the agent’s size as small as possible. ***During its operation, the skill can update or delete existing beliefs or create new ones.*** A skill operation may depend on beliefs created by other skills, and the *Skill Manager* is therefore in charge of dispatching asynchronously these beliefs to the interested skill in a

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

transparent way. It holds all the necessary information about the skills in the *Skill Base*.

Boudaoud at Sec. 3.1.

2444. Further examples include Boudaoud at Sections 3, 5.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2445. Boudaoud discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

“Nevertheless, different types of agents reflect a set of properties, which common among them and are described below [12]:

- ***Autonomy: is the ability of an agent to operate without direct intervention of humans or other agents and to have some kind of control based on its internal and/or external environments***

- Co-operation: an Agent is co-operative and is able to have a social ability. This sociability allows an agent to interact with other agents for the purpose of performing tasks that are beyond the capability of a particular agent. This capability goes from delegation (distribution of sub-tasks) to peer-to-peer interworking.

- Proactiveness: it is the agent’s ability to anticipate situations and change its course of action to avoid them. Proactive agents are capable of exhibiting goal-direct behaviors by taking some initiative [13][14].

- Reactivity: this kind of behavior means that the agent reacts in real-time to changes that occur in its environments.

- ***Adaptability: is the ability of an agent to modify its behavior over time to fulfill its problem-solving goal.***

- ***Intelligence: the term "Intelligence" means that the agent is able to exhibit a certain level of intelligence priority, ranging from predefined actions (planning) up to self-learning (define new actions).***

- ***Flexibility: is the ability an agent should have to adapt itself to cope with the environment in which it is situated.***

- Mobility: an Agent is mobile. It is capable of moving from one localisation to another in order to perform a particular task or to react to a particular event

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Having studied the properties of the [Intelligent Agent (IA)] and the aspects and requirements of a security management, it can be concluded that IA provides a more coherent and flexible approach of security management. The security management architecture based on the concept of IA can be conceived as if it were made of the autonomous

IAs co-operating with each other to achieve Global Security Policy. The section 5 describes the security management architecture.

Boudaoud at Sec. 2.

2446. Further examples include Boudaoud at Sections 2, 3, 5.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2447. Boudaoud discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

***[Distributed Intrusion Detection System (DIDS)] operates on a local area network (LAN)*** and its architecture combines distributed monitoring and data reduction with centralized data analysis [3]. A DIDS director, a LAN monitor, and a series of host monitor constitute it. The LAN monitor reports to the DIDS director unauthorized or suspicious activities on the network. The host monitors collect audit data for the individual host and perform some simple analysis on the data. The relevant information is then transmitted to the DIDS director. This director is responsible for analyzing all these data and detecting possible attacks. A shortcoming of DIDS is that the centralized nature of DIDS will limit its usefulness in wide area networks where communication with a central director from all hosts may swamp portions of the network.

***[Co-operating Security Managers (CSM)]*** was designed to perform intrusion detection in a distributed environment [4]. ***A CSM must be run on each computer connected to a network*** to facilitate the co-operative detection of network intrusions. It consists of following parts:

- a local intrusion detection component. It performs intrusion detection for the local host and is responsible for proactive detection of attacks on other host;
- a security manager which co-ordinates the distributed intrusion detection between CSMs;
- an intruder handling component. Its role is to take actions when an intruder is detected;
- a graphical user interface;

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- a command monitor which intercepts the commands executed by a user and sends for analysis;
- a TCP communication module.”

Boudaoud at Sec. 1.

2448. Further examples include Boudaoud at Sections 1, 3.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2449. Boudaoud discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2450. Boudaoud discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2451. Boudaoud discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2452. Boudaoud discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2453. Boudaoud discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

2454. Boudaoud discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2455. Boudaoud discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

2456. Boudaoud discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2457. Boudaoud discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2458. Boudaoud discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*assigned goal of the agent is expressed as a policy.*

2459. Boudaoud discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2460. Boudaoud discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2461. Boudaoud discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2462. Boudaoud discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2463. Boudaoud discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2464. Boudaoud discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2465. Boudaoud discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

2466. I expect to testify that Boudaoud anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Boudaoud discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-14.

**Q. Anticipation by and/or Obviousness in View of *Distributed Policy Based Management Enabling Policy Adaptation on Monitoring Using Active Network Technology*, by Yoshihara, et al., published in October 2001 at the 12th International Workshop on Distributed Systems.**

2467. As explained in detail below and in the chart attached as Ex. A-21, Yoshihara anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2468. Yoshihara discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In *policy-based management*, in addition to deliver and enforce policies in managed systems, it is inevitable to *manage policy life-cycle*. We mean the policy life-cycle as a sequence of processes involving monitoring to see if the enforced policies actually work at operators' will and adapting them based on monitoring. However, enabling such policy life-cycle management by the current centralized management paradigm such as SNMP may result in poor scalability and reliability. This is typically due to much bandwidth consumption for monitoring and communication failure between a manager and an agent. It may also burden the operators with a heavy load in analyzing management information for the policy adaptation. For a solution to that, we propose a *scalable and reliable policy-based management scheme enabling the policy life-cycle management using the active network technology*. In the scheme, we provide a new management script describing policies and also how their life-cycle should be managed, and execute the script on the managed systems called active nodes. The scheme can make the current policy-based management more scalable by reducing management traffic, more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden. We implement a prototype system based on the scheme adopting Differentiated Services as a policy enforcement mechanism, and evaluate the scheme from the following viewpoints: the advantage of policy adaptation on monitoring, the amount of management traffic required and the load on the managed systems executing the management scripts. We also discuss how the prototype system could be integrated with managed systems compliant with standards emerging in marketplace.

Yoshihara at 1.

2469. Further examples include Yoshihara at 2, 4; Yoshihara at Fig. 1.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2470. Yoshihara discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

In the scheme, we provide a new management script describing *policies* and also how their life-cycle should be managed, and *execute the script on the managed systems called active nodes*. The scheme can make the current policy-based management more scalable by reducing management traffic, more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden.

Yoshihara at 1.

2471. Further examples include Yoshihara at 1, 2, 3, 4, 5, 7, 9, 11; Yoshihara at Figs. 1, 2, 3, 11.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network;*

2472. Yoshihara discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

A network operator, first, decides a ***policy to be enforced on a managed system such as a router and a switch*** (Fig.1(1)).

Yoshihara at 2.

2473. Further examples include Yoshihara at 6, 7, 8-9.

d. *is capable of perceiving its own state*

2474. Yoshihara discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

The policy life-cycle means a sequence of processes involving ***monitoring*** to see if the enforced policies actually work at operators' will and adapting them based on the monitoring.

Yoshihara at 1.

2475. Further examples include Yoshihara at 4, 5, 6, 7, 8, 9.

e. *and is able to clone itself,*

2476. Yoshihara discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

A network operator, first, decides a ***policy to be enforced on a managed system such as a router and a switch*** (Fig.1(1)).

Yoshihara at 2.

2477. Further examples include Yoshihara at 5, 9.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2478. Yoshihara discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

In the scheme, we provide a new management script describing ***policies*** and also how their life-cycle should be managed, and ***execute the script on the managed systems called active nodes***. The scheme can make the current policy-based management more scalable by reducing management traffic,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden.

Yoshihara at 1.

2479. Further examples include Yoshihara at 2, 3, 5, 7; Yoshihara at Figs. 2, 3.

g. *monitoring the computer network;*

2480. Yoshihara discloses this element of claim 1, “monitoring the computer network.”

’730 Pat. at Cl. 1. For example:

The purpose of Diffserv is to establish peak boundaries on traffic values and the boundaries are given in the static forms. So, in order to make the most of limited network resources, a ***mechanism for adapting and optimizing the boundaries according to the dynamic nature of network utilization*** is required. Such a requirement is also stated in some early literatures [Wie94, Goh98]. In the policy-based management, therefore, it is inevitable to manage policy life-cycle management. We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) ***traffic monitoring*** to see if the enforced policy works at operators' will, and 3) policy adaptation (updating the condition and action clauses) based on the monitoring, as shown in Fig. 4.

Yoshihara at 4.

2481. Further examples include Yoshihara at 4, 5, 6, 7, 8.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2482. Yoshihara discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

The purpose of Diffserv is to establish peak boundaries on traffic values and the boundaries are given in the static forms. So, in order to make the most of limited network resources, a ***mechanism for adapting and optimizing the boundaries according to the dynamic nature of network utilization*** is required. Such a requirement is also stated in some early literatures [Wie94, Goh98]. In the policy-based management, therefore, it is inevitable to manage policy life-cycle management. We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) ***traffic monitoring*** to see if the enforced policy works at operators' will, and 3) ***policy adaptation (updating the condition and action clauses) based on the monitoring***, as shown in Fig. 4.

Yoshihara at 4.

2483. Further examples include Yoshihara at 4, 6, 7, 8; Yoshihara at Fig. 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- i. *including predicting a failure of a network component based on a prediction algorithm*

2484. Yoshihara discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

The purpose of Diffserv is to establish peak boundaries on traffic values and the boundaries are given in the static forms. So, in order to make the most of limited network resources, a ***mechanism for adapting and optimizing the boundaries according to the dynamic nature of network utilization*** is required. Such a requirement is also stated in some early literatures [Wie94, Goh98]. In the policy-based management, therefore, it is inevitable to manage policy life-cycle management. We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) ***traffic monitoring*** to see if the enforced policy works at operators' will, and 3) ***policy adaptation (updating the condition and action clauses) based on the monitoring***, as shown in Fig. 4.

Yoshihara at 4.

2485. Further examples include Yoshihara at 4, 6, 7, 8; Yoshihara at Fig. 4.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2486. Yoshihara discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

The purpose of Diffserv is to establish peak boundaries on traffic values and the boundaries are given in the static form. So, in order to make the most of limited network resources, a ***mechanism for adapting and optimizing the boundaries according to the dynamic nature of network utilization*** is required. Such a requirement is also stated in some early literatures [Wie94, Goh98]. In the policy-based management, therefore, it is inevitable to manage policy life-cycle management. We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) ***traffic monitoring*** to see if the enforced policy works at operators' will, and 3) ***policy adaptation (updating the condition and action clauses) based on the monitoring***, as shown in Fig. 4.

Yoshihara at 4.

2487. Further examples include Yoshihara at 4, 6, 7, 8; Yoshihara at Fig. 4.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2488. Yoshihara discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The purpose of Diffserv is to establish peak boundaries on traffic values and the boundaries are given in the static forms. So, in order to make the most of limited network resources, a ***mechanism for adapting and optimizing the boundaries*** according to the ***dynamic nature of network utilization*** is required. Such a requirement is also stated in some early literatures [Wie94, Goh98]. In the policy-based management, therefore, it is inevitable to manage policy life-cycle management. We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) traffic monitoring to see if the enforced policy works at operators' will, and 3) ***policy adaptation (updating the condition and action clauses)*** based on the monitoring, as shown in Fig. 4.

Yoshihara at 4.

2489. Further examples include Yoshihara at 4, 6, 7, 8, 9; Yoshihara at Fig. 4.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2490. Yoshihara discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

The policy may be requested from the management script within the managed system, a management system, or a surrogate host.

The policy life-cycle means a sequence of processes involving ***monitoring*** to see if the enforced policies actually work at operators' will and adapting them based on the monitoring.

Yoshihara at 1.

2491. Further examples include Yoshihara at 4, 5, 9; Yoshihara at Fig. 4.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2492. Yoshihara discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

In the scheme, we provide a new management script describing ***policies*** and also how their life-cycle should be managed, and ***execute the script on the managed systems called active nodes***. The scheme can make the current policy-based management more scalable by reducing management traffic, more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden.

Yoshihara at 1.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2493. Further examples include Yoshihara at 2, 3, 5, 7; Yoshihara at Figs. 2, 3.

**3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2494. Yoshihara discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

The policy life-cycle means a sequence of processes involving **monitoring** to see if the enforced policies actually work at operators' will and adapting them based on the monitoring.

Yoshihara at 1.

2495. Further examples include Yoshihara at 4, 5, 6, 7, 8, 9.

- b. *constructing a topological representation of the computer network from the information.*

2496. Yoshihara discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

For the purpose of policy life-cycle management, we introduce a new management script describing not only policies but also how to manage their life-cycle. As shown in Fig.6, it basically consists of the following three parts each mapped to the component of the policy life-cycle: the policy enforcement part, the traffic monitoring part, and the policy adaptation part.

Yoshihara at 4.

2497. Further examples include Yoshihara at 3, 5, 8; Yoshihara at Figs. 1, 10.

**4. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2498. Yoshihara discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

For the purpose of policy life-cycle management, we introduce a new management script describing not only policies but also how to manage their life-cycle. As shown in Fig.6, it basically consists of the following three parts each mapped to the component of the policy life-cycle: the policy enforcement part, the traffic monitoring part, and the policy adaptation part.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Yoshihara at 4.

2499. Further examples include Yoshihara at 3, 5, 8; Yoshihara at Figs. 1, 10.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2500. Yoshihara discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

A number of ***adaptation algorithms***, from simple ones to complex ones, may be possible. Below, we show three simple but practical adaptation algorithms each for the item to be adapted in Section 4.3.1, while an investigation of more suitable one is out of the scope of the paper and is left as a future work.

Yoshihara at 7-8.

6. ***Claim 7***

- a. *A computer network, comprising:*

2501. Yoshihara discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

In this paper, in the context of IETF policy-based management, we propose a new ***policy-based management scheme enabling the policy life-cycle management***. In the scheme, we introduce a new management script describing not only policies but also how their life-cycle should be managed. ***The script is executed on the managed system*** assumed to have enough computation resources, ***as well as in the active network technology [Pso99, RS00], the distributed management paradigm such as MbD (Management by Delegation) [YGY91, GY98], and the management by mobile software agents [ZHG99, GGG000]***. By executing the script on the managed system, the scheme can make the current policy-based management more scalable by reducing management traffic, more reliable by distributing management tasks to the managed systems, and more promising by alleviating the operators' burden.

Yoshihara at 2.

2502. Further examples include Yoshihara at 1, 4, 5; Yoshihara at Fig. 1.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*hardware*

2503. Yoshihara discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

2504. Yoshihara discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

2505. Yoshihara discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2506. Yoshihara discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2507. Yoshihara discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2508. Yoshihara discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

A network operator, first, decides a ***policy to be enforced on a managed system such as a router and a switch*** (Fig.1(1)).

Yoshihara at 2.

2509. Further examples include Yoshihara at 9, 11; Yoshihara at Figs. 1, 11.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*thereby to determine an optimal policy for the computer network*

2510. Yoshihara discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2511. Yoshihara discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2512. Yoshihara discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2513. Yoshihara discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2514. Yoshihara discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*an operational characteristic of the network.*

2515. Yoshihara discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2516. Yoshihara discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

We implement a prototype system based on the scheme adopting ***Differentiated Services [BBC+98] as the policy enforcement mechanism.***

Yoshihara at 2.

2517. Further examples include Yoshihara at 2-9; Yoshihara at Fig. 1.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2518. Yoshihara discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

IETF [IETa] and DMTF [DMT] jointly define information model for specifying a policy. ***IETF [IETa, IETb] also defines a framework of the policy-based management and protocols for policy delivery and enforcement.*** In the context of IETF policy-based management, a policy consists of a condition clause and an action clause applied only if the condition clause is evaluated to be true. There are two classes of policies with respect to their purposes: One is of QoS policies for priority and bandwidth control inducing priority queuing and packet shaping. Another is of security policies for access control inducing packet filters on a firewall machine and a WWW server. In this paper, we focus on the QoS policies hereafter. Figure I shows a framework of the current policy-based management defined by IETF

Yoshihara at 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2519. Further examples include Yoshihara at 2-6, 8.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2520. Yoshihara discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

A network operator, first, decides a ***policy to be enforced on a managed system such as a router and a switch*** (Fig.1(1)).

Yoshihara at 2.

2521. Further examples include Yoshihara at 9, 11; Yoshihara at Figs. 1, 11.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2522. Yoshihara discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

When a managed system receives the request (Fig. 1 0(4)), the ***execution environment in the managed system*** optionally loads the necessary class files (Fig. I 0(5)), and ***instantiates*** the management script. ***The management script then sets the provided policies to be enforced to the diffserv module*** (Fig. 1 0(6)), and responses to the management system if the enforcement has completed successfully or not (Fig. I 0(7)).

Yoshihara at 9.

2523. Further examples include Yoshihara at 2, 7, 11; Yoshihara at Fig. 11.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*selected from the group comprising of spawn, kill and suspend.*

2524. Yoshihara discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

When a managed system receives the request (Fig. 1 0(4)), the ***execution environment in the managed system*** optionally loads the necessary class files (Fig. I 0(5)), and ***instantiates*** the management script. The management script then sets the provided policies to be enforced to the diffserv module (Fig. 1 0(6)), and responses to the management system if the enforcement has completed successfully or not (Fig. I 0(7)).

Yoshihara at 9.

2525. Further examples include Yoshihara at 2, 3, 11; Yoshihara at Fig. 11.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2526. Yoshihara discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

When a managed system receives the request (Fig. 1 0(4)), the ***execution environment in the managed system*** optionally loads the necessary class files (Fig. I 0(5)), and ***instantiates*** the management script. The management script then sets the provided policies to be enforced to the diffserv module (Fig. 1 0(6)), and responses to the management system if the enforcement has completed successfully or not (Fig. I 0(7)).

Yoshihara at 9.

2527. Further examples include Yoshihara at 2, 7, 11; Yoshihara at Fig. 11.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2528. Yoshihara discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

***The policy can be stored in a directory server*** and may be reused using LDAP (Lightweight Directory Access Protocol) (Fig. I (2)).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Yoshihara at 2-3.

2529. Further examples include Yoshihara at 9.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2530. Yoshihara discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

8. The management script is automatically generated in the form of a Java object after providing the items described in Section 4.3. 1 via **GUI**.

Yoshihara at 8.

2531. Further examples include Yoshihara at 9.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2532. Yoshihara discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2533. Yoshihara discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The purpose of Diffserv is to establish peak boundaries on traffic values and the boundaries are given in the static forms. So, in order to make the most of limited network resources, a ***mechanism for adapting and optimizing the boundaries according to the dynamic nature of network utilization*** is required. Such a requirement is also stated in some early literatures [Wie94, Goh98]. In the policy-based management, therefore, it is inevitable to manage policy life-cycle management. We mean the policy life-cycle as a sequence of processes involving: 1) policy decision and enforcement, 2) ***traffic monitoring*** to see if the enforced policy works at operators' will, and



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3) *policy adaptation (updating the condition and action clauses) based on the monitoring*, as shown in Fig. 4.

Yoshihara at 4.

2534. Further examples include Yoshihara at 4, 6, 7, 8; Yoshihara at Fig. 4.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2535. Yoshihara discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

The policy life-cycle means a sequence of processes involving ***monitoring*** to see if the enforced policies actually work at operators' will and ***adapting*** them based on the monitoring.

Yoshihara at 1.

2536. Further examples include Yoshihara at 2, 4, 5, 9; Yoshihara at Figs. 4, 10.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2537. Yoshihara discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

A network operator, first, decides a ***policy to be enforced on a managed system such as a router and a switch*** (Fig.1(1)).

Yoshihara at 2.

2538. Further examples include Yoshihara at 9, 11; Yoshihara at Figs. 1, 11.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2539. Yoshihara discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *is able to communicate with other software agents in the computer network*

2540. Yoshihara discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2541. Yoshihara discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2542. Yoshihara discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2543. Yoshihara discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2544. Yoshihara discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2545. Yoshihara discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

2546. Yoshihara discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2547. Yoshihara discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2548. Yoshihara discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2549. Yoshihara discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2550. Yoshihara discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2551. Yoshihara discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2552. Yoshihara discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2553. Yoshihara discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2554. Yoshihara discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2555. Yoshihara discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

2556. I expect to testify that Yoshihara anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Yoshihara discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-21.

**R. Anticipation by and/or Obviousness in View of Cisco NetRanger, published by Wheelgroup/Cisco by August 1997 at the latest.**

2557. As explained in detail below and in the chart attached as Ex. A-8, Cisco NetRanger anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2558. Cisco NetRanger discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

NetRanger is an intrusion detection system that can be plugged into your TCP/IP networks in a variety of ways. It detects and responds to unauthorized activity in real time. Unauthorized activity includes attempts to circumvent an existing firewall, router, and network security policies, as well as misuse of authorized services. NetRanger is an enterprise-wide solution that can monitor a large number of networks via centralized management. As shown in Figure 1.1, NetRanger consists of three basic components: a Sensor, a communication system, and a Director.

2.1.1 User’s Guide at 1-1; *see also* id at 1-2.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2559. Cisco NetRanger discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

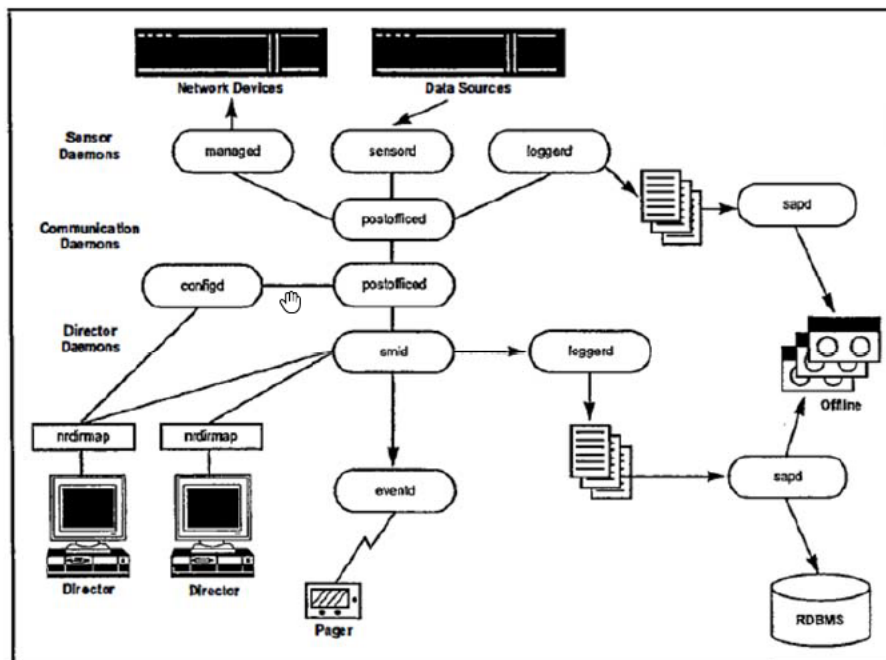
Each of NetRanger's subsystems (Sensor, Communication, Director) requires packaging of services. These are broken into their separate operational components, or daemons (see Figure 1.3). The NetRanger daemon that logs events is separate from the daemons that perform network sensing and device management. The results are speed, durability, scalability, and Independence.

Because each of NetRanger's daemon services is purpose-built for a specific task, it is easier to optimize each service without compromising the functionality of the others. Purpose-built components are also easier to debug and upgrade, and are more durable than multitasking systems.

2.1.1 User's Guide at 1-3; *see also id* at 1-4, 1-5, 1-6, 1-7, 1-13, H-4, H-5, H-6.

c. *is able to communicate with other software agents in the computer network;*

2560. Cisco NetRanger discloses this element of claim 1, "is able to communicate with other software agents in the computer network." '730 Pat. at Cl. 1. For example:



**Figure 1.3: NetRanger 2.1.1 Architecture**

2.1.1 User's Guide at 1-4

The Post Office subsystem provides the communication backbone for remote monitoring and management of the Sensor network device. All

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

communication is supported by a proprietary, connection-based protocol that can switch between alternate routes to maintain the point-to-point connections specified in its routing tables. All messages are routed based on a three-part address that includes organization, host, and application identifiers.

2.1.1 User's Guide at 1-4; see also *id.* at 1-5, 1-10, 1-11, 1-12.

d. *is capable of perceiving its own state*

2561. Cisco NetRanger discloses this element of claim 1, "is capable of perceiving its own state." '730 Pat. at Cl. 1. For example:

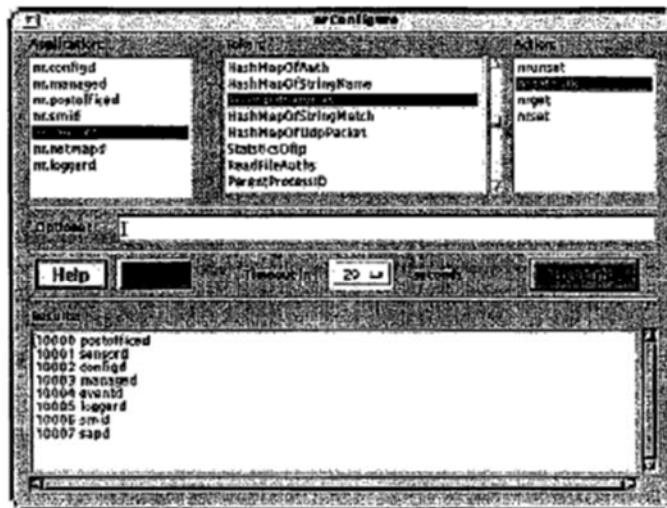


Figure 1.7: Sensor Configuration Interface

The Applications displayed in nrConfigure's top left-hand list box represent the services running on the first Sensor icon highlighted by the user. The top center list box displays all of the Tokens that apply to the Application currently highlighted in the left-hand list box. The top right-hand list box displays the Actions that are allowed for the currently selected Token. Figure 1.7 shows the Actions and Tokens for the sensord daemon.

2.1.1 User's Guide at 1-14; see also *id.* at 1-4, 1-17, 1-10.

e. *and is able to clone itself,*

2562. Cisco NetRanger discloses this element of claim 1, "and is able to clone itself." '730 Pat. at Cl. 1. For example:

This section lists the default NetRanger alarm settings. These settings provide for nearmaximum intrusion detection but are not configured to provide automatic response. For each of the ID signatures, string matches,



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

transport events, and policy violations, you can turn on auto-logging or auto-shunning by editing `sensord.conf` or changing the settings through the `nrConfigure` GUI. The default levels will give you some idea of the usual level of threat for each of these items.

2.1.1 User's Guide at C-4; see also *id.* at 1-7, 1-19, 4-17, 4-57.

The default or user-defined configuration for a sensor/director may be applied to other sensors/directors. Under NetFuel's application of the claims, NetRanger satisfies this limitation.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2563. Under NetFuel's apparent application of the claims in its infringement contentions, Cisco NetRanger discloses this element of claim 1, "and wherein the goal is a programmatic expression of a predefined task for the software agent." '730 Pat. at Cl. 1. For example:

There are two ways to change the configuration of NetRanger daemons running on remote Sensor machines.

1. You can use the `nrgetlnrset` infrastructure to modify daemon characteristics based on "tokens" that are specified by the user. These commands can be run from the command line of the Director machine.

2. You can also use the graphical browser utility that comes with the Director.

To bring up the utility, select one or more Application or Machine symbols, and then select the menu option Security→Configure. To learn more about `nrConfigure`, see the `nrConfigure` section.

2.1.1 User's Guide at 4-23; see also *id.* at 4-49, H-4, H-5.

g. *monitoring the computer network;*

2564. Under NetFuel's apparent application of the claims in its infringement contentions, Cisco NetRanger discloses this element of claim 1, "monitoring the computer network." '730 Pat. at Cl. 1. For example:

The Sensor handles real-time intrusion detection and device management. It uses a rules-based engine to distill large volumes of IP network traffic into meaningful events. A Sensor can either capture network traffic directly from the network or receive it from a network device, such as an STK packet filter firewall.

The Packet Filtering Device is a router or firewall residing on the network at a point of entry to other networks. The Sensor can reconfigure these



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

devices on the fly to shun the source of an attack. Many of these devices can also serve as the data source for the intrusion Detection subsystem. Device management is optional because there are places within networks where detection is all that is required-e.g., at connections between internal networks.

2.1.1 User's Guide at 1-4; see also *id.* at 1-5, 1-7, 1-13, 4-95.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2565. Under NetFuel's apparent application of the claims in its infringement contentions, Cisco NetRanger discloses this element of claim 1, "creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network." '730 Pat. at Cl. 1. For example:

The exec command instructs an application to execute an action external to itself. For example, you can instruct a Sensor to shun a specific IP address by issuing an exec against managed, which in turn modifies the filter configuration on its packet filter. The exec command is also used for such tasks as writing configuration information to disk.

2.1.1 User's Guide at 1-15.

The Director DMP capability also analyzes data. Rather than locking the user into a single tool on the Director machine, this task is better served by 3rd-party tools on a separate Windows platform, such as the IO Objects® report writer from IO Software and multi-dimension analysis tools such as PowerPiav® from Cognos. Trouble-ticketing systems such as Remedy's Action Request System® (ARS) can also be implemented on top of NetRanger's alarm data.

2.1.1 User's Guide at 1-18; see also *id.* at 1-7, 1-9, 1-19, 2-3, 2-4, 2-6, 2-7, 2-8, 4-56, 4-94, C-9, C-10, C-11, E-12, E-13, H-3;

#### IDS Overview<sup>11</sup>

- i. *including predicting a failure of a network component based on a prediction algorithm*

2566. Cisco NetRanger discloses this element of claim 1, "including predicting a failure of a network component based on a prediction algorithm." '730 Pat. at Cl. 1. For example:

<sup>11</sup> Cisco, Systems, Inc., NetRanger Intrusion Detection System Technical Overview (1998), [http://storage.library.opu.ua/online/external/cisco/products/778/security/netranger/ntran\\_tc.htm](http://storage.library.opu.ua/online/external/cisco/products/778/security/netranger/ntran_tc.htm) ("IDS Overview")

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

NetRanger is not only an Intrusion Detection system, it also looks for patterns of misuse, or attack signatures. Patterns can be as simple as an attempt to access a specific port on a specific host, or as complex as sequences of operations distributed across multiple hosts over an arbitrary period of time. The first type of pattern is termed an atomic signature; the second, a composite signature. Often, NetRanger suspicious activity and alert security personnel long before an attack occurs.

NetRanger searches for patterns of misuse by examining either the data portion or the header portion of network packets. Content-based attack signatures derive from the data portion, and context-based signatures derive from the header portion. The table on the below illustrates the types and combinations of signatures that NetRanger detects.

2.1.1 User's Guide at 1-6, 1-7; see also *id.* at 4-62 to 4-96.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2567. Cisco NetRanger discloses this element of claim 1, "wherein said modeling comprises determining appropriate policy based on the prediction." '730 Pat. at Cl. 1. For example:

In addition to displaying and logging alarm events, the Director can generate user-defined actions via eventd. A typical action might be to generate pager notifications via e-mail messages or feed data onto 3rd-party devices, such as a printer. Support for multiple action scripts is also provided. While eventd makes no distinction between alarm types and levels, the default action script shipped with this service shows how actions can be triggered based on these criteria.

2.1.1 User's Guide at 1-19; see also *id.* at 2-7, 2-8, 4-62 to 4-96, 5-36, C-4, E-11

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2568. Cisco NetRanger discloses this element of claim 1, "dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy." '730 Pat. at Cl. 1. For example:

A major difference between a Sensor configured to work with a Cisco router and a Sensor configured to work with a StorageTek router is the fact that a Sensor cannot detect when a packet has been failed by the Access Control Lists (ACLs) on a Cisco router. This is because the Sensor is capturing data behind the router and cannot see any of the denied traffic. Sensors connected to StorageTek routers receive notifications for this type of activity because the NetSentry operating system on these devices is able to detect and report these type of events. It should be noted however, that a Sensor's shunning

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

capability allows it to dynamically update a router's security access policy in response to unauthorized events that do get through.

#### 2.1.1 User's Guide at C-9.

A network device must sit within the flow of traffic within a network. This is accomplished by the network device acting as either a router or a bridge. A facility must support the selective copying of certain packets (TCP/IP, IPX/SPX, etc.) to the Sensor for analysis. The device must also support the capability to be dynamically reconfigured to enforce a security policy. The BorderGuard and Passport provide this through the use of NSG's NetSentry.

2.1.1 User's Guide at G-2; see also *id.* at 1-9, 2-6, 2-8, 3-344-94; Managing Cisco Network Security at 425<sup>12</sup>.

#### IDS Overview

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2569. Cisco NetRanger discloses this element of claim 1, "wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task." '730 Pat. at Cl. 1. For example:

A major difference between a Sensor configured to work with a Cisco router and a Sensor configured to work with a StorageTek router is the fact that a Sensor cannot detect when a packet has been failed by the Access Control Lists (ACLs) on a Cisco router. This is because the Sensor is capturing data behind the router and cannot see any of the denied traffic. Sensors connected to StorageTek routers receive notifications for this type of activity because the NetSentry operating system on these devices is able to detect and report these type of events. It should be noted however, that a Sensor's shunning capability allows it to dynamically update a router's security access policy in response to unauthorized events that do get through.

2.1.1 User's Guide at C-9; see also *id.* at C-2, H-3, 1-7, 1-8, 1-9, 2-6, 2-8, 3-34, 4-1, 4-24, 4-27, 4-56, 4-57, 4-94, 5-3; Managing Cisco Network Security at 425.

#### IDS Overview

#### 2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is*

<sup>12</sup> Russell Lusignan, et. al., Managing Cisco Network Security: Building Rock-Solid Networks (Florent Parent, et. al. eds., Syngress Publishing, Inc., 2000) ("Managing Cisco Network Security")

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*expressed as a policy.*

2570. Cisco NetRanger discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

There are two ways to change the configuration of NetRanger daemons running on remote Sensor machines.

1 . You can use the nrgetlnrset infrastructure to modify daemon characteristics based on "tokens" that are specified by the user. These commands can be run from the command line of the Director machine.

2. You can also use the graphical browser utility that comes with the Director.

To bring up the utility, select one or more Application or Machine symbols, and then select the menu option Security→Configure. To learn more about nrConfigure, see the nrConfigure section.

#### 2.1.1 User's Guide at 4-23

The nrConfigure GUI works in the following manner. When you press Execute for a given application, token, and action, a packet is sent to configd, which sends it on to postofficed. postofficed sends it on to the proper daemon, which processes the requests and replies. The reply returns through postofficed to configd and appears in the nrConfigure's Results window.

2.1.1 User's Guide at 4-49; see also *id.* at 4-50, H-4, H-5.

### 3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2571. Cisco NetRanger discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

The Sensor's capabilities are best described as network sensing, attack response, and device management, implemented respectively by the sensord, loggerd, and managed daemons diagrammed in Figure 1 .3.

#### Network Sensing

Network sensing is the primary function of NetRanger. To accomplish this task, the Sensor must pull data from the network for analysis. The Sensor can either receive its data directly from a managed network device or it can pull the data directly from the network itself. This data is reassembled and compared against a rule set indicating typical intrusion activity.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2.1.1 User's Guide at 1-5; see also *id.* at 1-15, 1-16, 4-1, 4-6, 4-7, 4-8, 4-22, 4-42, 4-69, 4-70, 4-71.

- b. *constructing a topological representation of the computer network from the information.*

2572. Cisco NetRanger discloses this element of claim 3, "constructing a topological representation of the computer network from the information." '730 Pat. at Cl. 3. For example:

The fifth question asks if you want alarm icons to be drawn on the IPMap. This option has no effect with this release. It may be used in a future version. The IPMap is the submap hierarchy created by the ipmap application. The ipmap application is the part of OpenView/NetView that draws a picture of the IP Topology. The advantage of having alarms represented on the IPMap is that you can view Fault status and Security status on the same screen. The disadvantage is that performance is degraded because extra icons are being created.

2.1.1 User's Guide at 4-22.

The Director's most prominent capability is display of real-time event information, which is based on the smid daemon, the nrdirmap application, and a supported network management package (currently Open View and NetView). The smid daemon translates Sensor-generated event records and translates them into a format that nrdirmap understands. The nrdirmap application uses the OVW API (Open View Windows Application Programming Interface) to tell the OpenView user interface what security information to present to the user. Security information is presented via icons drawn on one or more network security maps. Figure 1.8 shows an example of the Director's display of a collection of Sensor maps.

2.1.1 User's Guide at 1-15; see also *id.* at 1-16, 4-6, 4-7, 4-8, 4-42.

**4. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2573. Cisco NetRanger discloses claim 4, "[t]he method of claim 1, wherein the modeling uses a numerical method." '730 Pat. at Cl. 4. For example:

It would have been obvious to combine NetRanger with the OSPF routing protocol (RFC 2328). NetRanger monitors ICMP messages that change the routing table, and it would have been obvious to use NetRanger on a system that uses the OSPF routing protocol, and use OSPF state changes to trigger alarms. OSPF routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement the NetRanger system on network devices using OSPF.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2574. Cisco NetRanger discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

It would have been obvious to combine NetRanger with the OSPF routing protocol (RFC 2328). NetRanger monitors ICMP messages that change the routing table, and it would have been obvious to use NetRanger on a system that uses the OSPF routing protocol, and use OSPF state changes to trigger alarms. OSPF routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement the NetRanger system on network devices using OSPF.

6. ***Claim 7***

- a. *A computer network, comprising:*

2575. Cisco NetRanger discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

The Sensor handles real-time intrusion detection and device management. It uses a rules-based engine to distill large volumes of IP network traffic into meaningful events. A Sensor can either capture network traffic directly from the network or receive it from a network device, such as an STK packet filter firewall.

The Packet Filtering Device is a router or firewall residing on the network at a point of entry to other networks. The Sensor can reconfigure these devices on the fly to shun the source of an attack. Many of these devices can also serve as the data source for the intrusion Detection subsystem. Device management is optional because there are places within networks where detection is all that is required-e.g., at connections between internal networks.

2.1.1 User’s Guide at 1-4

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2576. Cisco NetRanger discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

2577. Cisco NetRanger discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

2578. Cisco NetRanger discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2579. Cisco NetRanger discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

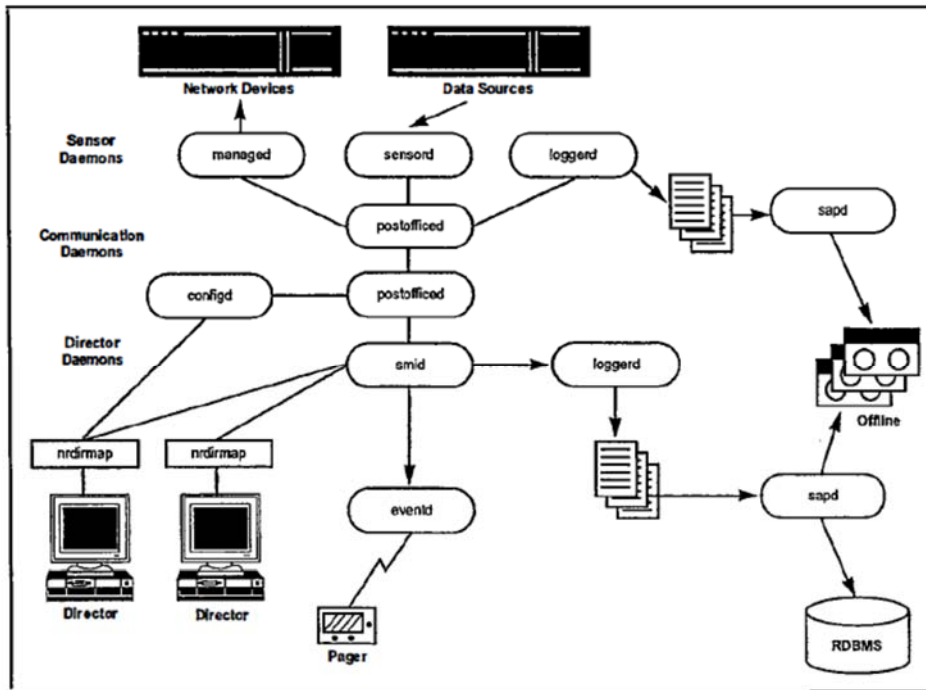
2580. Cisco NetRanger discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

2581. Cisco NetRanger discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:



**Figure 1.3: NetRanger 2.1.1 Architecture**

2.1.1 User's Guide at 1-4

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2582. Cisco NetRanger discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*of a network component*

2583. Cisco NetRanger discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the prediction;*

2584. Cisco NetRanger discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2585. Cisco NetRanger discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2586. Cisco NetRanger discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2587. Cisco NetRanger discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2588. Cisco NetRanger discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

The Post Office subsystem provides the communication backbone for remote monitoring and management of the Sensor network device. All communication is supported by a proprietary, connection-based protocol that can switch between alternate routes to maintain the point-to-point connections specified in its routing tables. All messages are routed based on a three-part address that includes organization, host, and application identifiers.

2.1.1 User’s Guide at 1-4; see also *id.* at 1-5, 1-10, 1-11, 1-12.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2589. Cisco NetRanger discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl.

12. For example:

The Post Office subsystem provides the communication backbone for remote monitoring and management of the Sensor network device. All communication is supported by a proprietary, connection-based protocol that can switch between alternate routes to maintain the point-to-point connections specified in its routing tables. All messages are routed based on a three-part address that includes organization, host, and application identifiers.

2.1.1 User’s Guide at 1-4; see also *id.* at 1-5, 1-10, 1-11, 1-12.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2590. Cisco NetRanger discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

“The NetRanger CD contains software to configure and install a NetRanger Sensor on either a x86 or S PARC Solaris workstation.

Minimum Requirements

Before installing a new NetRanger Sensor, prepare the workstation by installing the Solaris OS. Do not install the NetRanger Sensor software on a workstation that does not meet the minimum requirements outlined in the Minimum Requirements section of the NetRanger OS Installation Instructions.

2.1.1 User's Guide at 3-8, 3-14.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2591. Cisco NetRanger discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

The NetRanger CD contains software to configure and install a NetRanger Sensor on either a x86 or S PARC Solaris workstation.

Minimum Requirements

Before installing a new NetRanger Sensor, prepare the workstation by installing the Solaris OS. Do not install the NetRanger Sensor software on a workstation that does not meet the minimum requirements outlined in the Minimum Requirements section of the NetRanger OS Installation Instructions.

2.1.1 User's Guide at 3-8, 3-14.

Configuring the NetRanger Background Processes

The NetRanger background process configuration files enable the Director to communicate with Sensors on your network. Run the nrconfig utility to configure these files. (Please refer to the NetRanger Configuration Instructions and Worksheets section in this chapter for information about the nrconfig utility.)

1 . Stop the NetRanger/Director daemons by typing (as user netrangr)

/usr/nr/bin/nrstop3-27

2.1.1 User's Guide at 3-27.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2592. Cisco NetRanger discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl.

18. For example:

**Configuring the NetRanger Background Processes**

The NetRanger background process configuration files enable the Director to communicate with Sensors on your network. Run the nrconfig utility to configure these files. (Please refer to the NetRanger Configuration Instructions and Worksheets section in this chapter for information about the nrconfig utility.)

- 1 . Stop the NetRanger/Director daemons by typing (as user netrangr)

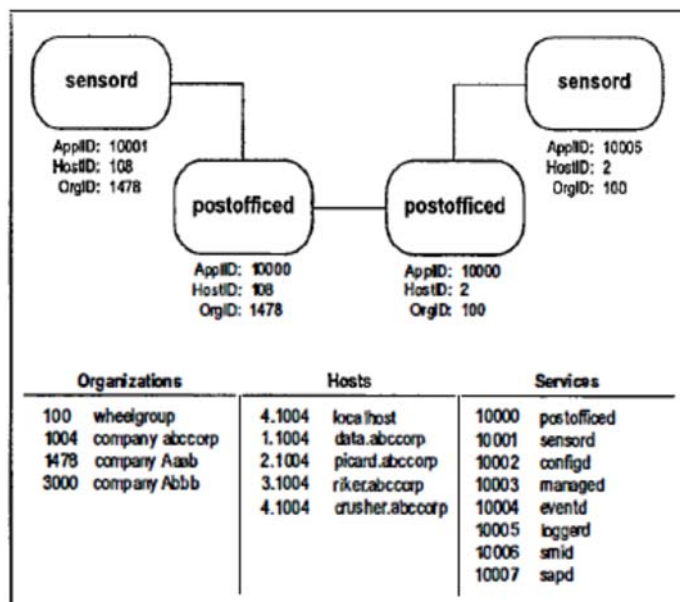
/usr/nr/bin/nrstop3-27

2.1.1 User’s Guide at 3-27.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2593. Cisco NetRanger discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1.5: Example /usr/nr/etc/ communication files**

## 2.1.1 User Guide at 1-10

NetFuel contends this limitation is inherent and thus is met by NetRanger under its interpretation of the claims.

Additionally, this limitation is rendered obvious in view of Kasteleijn. See Ex. A-2 at limitation 19.0.”

14. **Claim 21**

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2594. Cisco NetRanger discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

If your Sensor is in stand-alone mode (i.e., not working with a BorderGuard or Passport), then use a text editor to open the /usr/nr/etc/packetd.conf file. Otherwise, you will have to edit the /usr/nr/etc/sensord.conf file.

## 2.1.1 User’s Guide at 4-94.

## Default NetRanger Alarm Settings

This section lists the default NetRanger alarm settings. These settings provide for nearmaximum intrusion detection but are not configured to provide automatic response. For each of the ID signatures, string matches,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

transport events, and policy violations, you can turn on auto-logging or auto-shunning by editing sensord.conf or changing the settings through the nrConfigure GUI. The default levels will give you some idea of the usual level of threat for each of these items.

2.1.1 User's Guide at C-4, *see also id.* 4-2, 4-59, B-15, C-4, E-5, E-8, E-10.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2595. Cisco NetRanger discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

“The NetRanger Director is the Graphical User Interface (GUI) for the NetRanger system. The NetRanger Director (also called "the Director")

- provides a graphical, intuitive display of information pertaining to network security violations in real time;
- displays a hierarchical map of the remote NetRanger software and hardware (e.g., the Sensor processes and the Sensor hardware) that send security notifications to the Director;
- provides utilities for configuration of the remote NetRanger applications; and
- provides utilities to query the database of historical security events.

2.1.1 User's Guide at 4-1; *see also id.* at 4-49.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2596. Cisco NetRanger discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

2597. Cisco NetRanger discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The NetSentry-based devices have a copy\_to or log\_to feature that allows them to pass along activity that it fails at its filter point. In this way, the Sensor can detect and alarm on activity that never enters the network. This information is very valuable, because it may alert security personnel of an attempt to break through the network perimeter. If the policy violations continue, then the Sensor can dynamically instruct the packet filter to reject all traffic from the abusive host, or even from its entire network.

The Cisco-series routers do not support copying of failed packet activity to the Sensor. Instead, the Cisco device must allow all traffic to flow to a firewall, which implements policy filtering. The Sensor captures packets between the Cisco and the firewall, and can use the Cisco to dynamically shun unauthorized activity.

2.1.1 User’s Guide at 2-8, *see also id.* at 2-6, 2-8, B-6, G-2, H-3.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2598. Cisco NetRanger discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

“The Sensor handles real-time intrusion detection and device management. It uses a rules-based engine to distill large volumes of IP network traffic into meaningful events. A Sensor can either capture network traffic directly from the network or receive it from a network device, such as an STK packet filter firewall.

The Packet Filtering Device is a router or firewall residing on the network at a point of entry to other networks. The Sensor can reconfigure these devices on the fly to shun the source of an attack. Many of these devices can also serve as the data source for the intrusion Detection subsystem. Device management is optional because there are places within networks where detection is all that is required-e.g., at connections between internal networks.”

2.1.1 User’s Guide at 1-4; *see also id.* at 1-5, 1-6, 1-7, 1-13, 4-95.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*which when executed by a processor causes the processor to perform a method comprising*

2599. Cisco NetRanger discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

The NetRanger CD contains software to configure and install a NetRanger Sensor on either a x86 or S PARC Solaris workstation.

Minimum Requirements

Before installing a new NetRanger Sensor, prepare the workstation by installing the Solaris OS. Do not install the NetRanger Sensor software on a workstation that does not meet the minimum requirements outlined in the Minimum Requirements section of the NetRanger OS Installation Instructions.

2.1.1 User’s Guide at 3-8, 3-14.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2600. Cisco NetRanger discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2601. Cisco NetRanger discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2602. Cisco NetRanger discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2603. Cisco NetRanger discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy when it lacks an ability to perform the predefined task*

2604. Cisco NetRanger discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2605. Cisco NetRanger discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2606. Cisco NetRanger discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a predictive algorithm;*

2607. Cisco NetRanger discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2608. Cisco NetRanger discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

2609. Cisco NetRanger discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2610. Cisco NetRanger discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2611. Cisco NetRanger discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2612. Cisco NetRanger discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task;*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*and*

2613. Cisco NetRanger discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2614. Cisco NetRanger discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2615. Cisco NetRanger discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2616. Cisco NetRanger discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

**S. Anticipation by and/or Obviousness in View of the Cisco Catalyst 5000**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**Switch Family, marketed by Cisco by June 1995 at the latest.**

2617. As explained in detail below and in the chart attached as Ex. A-9, Cisco Catalyst 5000 anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

2618. Cisco Catalyst 5000 discloses the preamble of claim 1, "[a] method of managing a computer network, comprising." '730 Pat. at Cl. 1. For example:

**Unique Traffic Management Capabilities**

Embedded traffic management functionality is a critical component of the Catalyst 5000 design. The Catalyst 5000 supports extensive input buffers across its switching modules that function with buffer management software to eliminate packet loss during peak traffic periods.

The Catalyst 5000 architecture supports multiple levels of data prioritization, with each interface separately user-configurable as high or low priority. Maintenance of separate logical queues for each priority class reduces buffering delays, a critical issue with typically bursty network traffic and delay-sensitive traffic types such as voice and video.

The Catalyst 5000 family supports switched port analyzer software. This software gives network managers the ability to track traffic on any one of the 5000's switching ports.

Cisco is the only vendor to support embedded remote monitoring (RMON) across all of its switching platforms. Because RMON is fully integrated with the CiscoView device management application, network managers can view, configure and set alarms on a graphical representation of the switch.

Catalyst 5000 also supports RFC 1271 standard-compliant RMON across all ports, enabling users to centrally manage, administer and control traffic monitoring for all switched LAN segments simultaneously. Offering centralized control that greatly reduces administrative overhead, the embedded RMON agent can save the user up to 80 percent of the cost of using specialized hardware probes on each LAN segment.

"Cisco Delivers Switching for Wiring Closets" March 28, 1995

b. *assigning a goal to a software [agent], wherein the software agent*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*has its own runtime environment*

2619. Cisco Catalyst 5000 discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl.

1. For example:

Standard ACLs are the oldest type of ACL. They date back to as early as Cisco IOS Software Release 8.3. Standard ACLs control traffic by the comparison of the source address of the IP packets to the addresses configured in the ACL.

This is the command syntax format of a standard ACL.

access-list access-List-number {permit | deny}

{host/ source source-wildcard | any}

CSI-NF-00000013 at 8

Extended ACLs

Extended ACLs were introduced in Cisco IOS Software Release 8.3. Extended ACLs control traffic by the comparison of the source and destination addresses of the IP packets to the addresses configured in the ACL.

This is the command syntax format of extended ACLs. Lines are wrapped here for spacing considerations.

CSI-NF-00000013 at 8; *see also id.* at 9, 10.

Policing and Shaping Overview

Cisco IOS QoS offers two kinds of traffic regulation mechanisms: the rate-limiting feature of committed access rate (CAR) for policing traffic, and Generic Traffic Shaping (GTS) and Frame Relay Traffic Shaping (FRTS) for shaping traffic. You can deploy these features throughout your network to ensure that a packet, or data source, adheres to a stipulated contract and to determine the QoS to render the packet. Both policing and shaping mechanisms use the traffic descriptor for a packet—indicated by the packet's classification—to ensure adherence and service. (See the chapter "Classification Overview" for a description of a traffic descriptor.)

IOS 12.0 Policing and Shaping Overview;<sup>13</sup> *see also* Catalyst 5K Family RSM/VIP2 Cisco IOS 12.0 Release Notes.

To the extent the ACL or QoS software in Cisco IOS 12.0 on the Catalyst 5000 Switch Family fails to explicitly or implicitly disclose a “software

<sup>13</sup>([https://web.archive.org/web/20010203115400/http://www.cisco.com:80/univercd/cc/td/doc/product/software/ios120/12cgcr/qos\\_c/qcpolts.htm](https://web.archive.org/web/20010203115400/http://www.cisco.com:80/univercd/cc/td/doc/product/software/ios120/12cgcr/qos_c/qcpolts.htm))

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agent”, it would have been obvious to in view of Kasteleijn. *See* Ex. A-2, limitation 1.1. It would have been obvious to implement QoS or ACL software as a software agent to obtain the benefits of the MCs in Kasteleijn, for example, to maintain state during a resource crash or migration, and to improve resource reuse and reduce resource consumption.

- c. *is able to communicate with other software agents in the computer network;*

2620. Cisco Catalyst 5000 discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

The ACL/QoS software may be distributed among different line cards and controlled by a supervisor module in the Catalyst 5000 Switch Family. Under NetFuel’s interpretation of this claim, one ACL/QoS software agent may “communicate directly with another” ACL/QoS.

Catalyst 5505 The Catalyst 5505 is a high-performance, 5-slot chassis for the evolving Catalyst 5500 Series. It combines the size of the original Catalyst 5000 with the performance boost and added features of the Catalyst 5500. The Catalyst 5505 is ideal for high-performance wiring closets and data applications. The Catalyst 5505 supports Cisco’s Investment Protection standards because it can redeploy all the line cards in an existing Catalyst 5000 series. The Catalyst 5505 can be configured for backbone applications with feature-rich, scalable 100/1000 Ethernet, ATM, and FDDI, as well as optional redundant Supervisor Engines and power supplies. In the wiring closet, switched 10/100 Ethernet, Token Ring, and ATM modules provide high-performance connectivity.

Cisco 5000 Family – Multilayer Switches”, June 2000 at 2<sup>14</sup>;

*see also* IOS 12.0 Configuring IP Services; IOS 12.0 Quality of Service Commands.

To the extent this limitation is not disclosed by the Cisco Catalyst 5000 Switch Family, it is rendered obvious in combination with Kasteleijn.

*See* Ex. A-2, limitation 1.3.

- d. *is capable of perceiving its own state*

2621. Cisco Catalyst 5000 discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

Under NetFuel’s interpretation, an ACL or OoS agent “is at least aware of the state of its own execution with respect to IOS software process or subsystem activity.

<sup>14</sup> (<https://www.andovercg.com/datasheets/cisco-5000-5500-series-switches.pdf>)

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

To the extent this limitation is not disclosed by the Cisco Catalyst 5000 Switch Family, it is rendered obvious in combination with Kasteleijn.

See Ex. A-2, limitation 1.4.

e. *and is able to clone itself,*

2622. Cisco Catalyst 5000 discloses this element of claim 1, “and is able to clone itself.”

’730 Pat. at Cl. 1. For example:

**ip access-group**

To control access to an interface, use the **ip access-group** interface configuration command. To remove the specified access group, use the **no** form of this command.

**ip access-group** {*access-list-number* | *name*} {**in** | **out**}

**no ip access-group** {*access-list-number* | *name*} {**in** | **out**}

**Syntax Description**

|                           |  |
|---------------------------|--|
| <i>access-list-number</i> | Number of an access list. This is a decimal number from 1 to 199 or from 1300 to 2699. |
| <i>name</i>               | Name of an IP access list as specified by an <b>ip access-list</b> command.            |
| <b>in</b>                 | Filters on inbound packets.  |
| <b>out</b>                | Filters on outbound packets.   |

IOS 12.0 Configuring IP Services, *see also* IOS 12.0 Quality of Service Commands.

To the extent this limitation is not disclosed by the Cisco Catalyst 5000 Switch Family, it is rendered obvious in combination with Kasteleijn. See Ex. A-2, limitation 1.4.

f. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent; and*

2623. Cisco Catalyst 5000 discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

**ip access-group**

To control access to an interface, use the **ip access-group** interface configuration command. To remove the specified access group, use the **no** form of this command.

**ip access-group** {*access-list-number* | *name*} {**in** | **out**}

**no ip access-group** {*access-list-number* | *name*} {**in** | **out**}

**Syntax Description**

|                           |  |
|---------------------------|--|
| <i>access-list-number</i> | Number of an access list. This is a decimal number from 1 to 199 or from 1300 to 2699. |
| <i>name</i>               | Name of an IP access list as specified by an <b>ip access-list</b> command.            |
| <b>in</b>                 | Filters on inbound packets.  |
| <b>out</b>                | Filters on outbound packets.   |

2624. IOS 12.0 Configuring IP Services, *see also* IOS 12.0 Quality of Service Commands.

2625. To the extent this limitation is not disclosed by the Cisco Catalyst 5000 Switch Family, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.5.

g. *monitoring the computer network;*

2626. Cisco Catalyst 5000 discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

2627. Filtering via an ACL or performing QoS “monitors the network” under NetFuel’s application of the claims.

2628. To the extent this limitation is not disclosed by the Cisco Catalyst 5000 Switch Family, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.6.

2629. To the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger. A person of ordinary skill in the art would be motivated (as discussed in the NetRanger manual) to use Cisco Catalyst 5000 Switch Family devices as packet filtering devices monitored by NetRanger sensors for intrusion detection.

h. *creating test policy and modeling a behavior of the computer*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network based on the test policy to determine an optimal policy for the computer network,*

2630. Cisco Catalyst 5000 discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

Applying ACLs or QoS policies to interfaces satisfies this limitation under NetFuel’s application of the claims.

To the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger. A person of ordinary skill in the art would be motivated (as discussed in the NetRanger manual) to use Cisco Catalyst 5000 Switch Family devices as packet filtering devices monitored by NetRanger sensors for intrusion detection. *See*, Ex. A-8, limitation 1.7. In the combined system, ACL rules and policies, which are designed to filter, block, or otherwise rate-limit traffic, are test policies. Each test policy has values that are set, for example, for permitting or denying traffic or for use in OoS matching, and are set by default or modified or configured by a user. The test policies indicate how the traffic in the network should look. For example, particular traffic should not be permitted at all on certain ports, or the flow rates for a particular packet types should be at or below a specific threshold.

Execution and enforcement of ACL or OoS test policies then models and reflects the actual behavior of the traffic flow on the computer network which can be monitored by NetRanger to determine and then instantiate a revised, optimal policy based on certain triggering events.

For example, when a policy violation is detected by NetRanger, the Director determines the specifics of a particular event that is triggered to select a specific policy, which may include shunning the traffic by reconfiguring the Catalyst 5000 Switch Family’s ACLs to deny that traffic.

To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that will not result in failure), where modeling includes predicting failure of a network component (e.g. faults, inconsistencies or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings QoS and ACL agents), as taught by Goldman.

- i. *including predicting a failure of a network component based on a prediction algorithm*

2631. Cisco Catalyst 5000 discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The combination of Catalyst 5000 Switch Family and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 1.8.

To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that will not result in failure), where modeling includes predicting failure of a network component (e.g. faults, inconsistencies or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings QoS and ACL agents), as taught by Goldman.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2632. Cisco Catalyst 5000 discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

The combination of Catalyst 5000 Switch Family and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 1.9.

To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that will not result in failure), where modeling includes predicting failure of a network component (e.g. faults, inconsistencies or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings QoS and ACL agents), as taught by Goldman.

Furthermore, this limitation would have been obvious in view of de Rocha. *See* Ex. A-6, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. ifaults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new ACL or QoS settings), as taught by da Rocha.

- k. *dynamically modifying the assigned goal of the software agent by*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*replacing the assigned goal based on the optimal policy*

2633. Cisco Catalyst 5000 discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.”

’730 Pat. at Cl. 1. For example:

The combination of Catalyst 5000 Switch Family and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 1.10. As discussed above, the NetRanger system determines an appropriate action for a policy violation which may include reconfiguring the Catalyst 5000 Switch Family’s ACLs.

To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that will not result in failure), where modeling includes predicting failure of a network component (e.g. faults, inconsistencies or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings QoS and ACL agents), as taught by Goldman.

Furthermore, this limitation would have been obvious in view of de Rocha. *See* Ex. A-6, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. ifaults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new ACL or QoS settings), as taught by da Rocha.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2634. Cisco Catalyst 5000 discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

#### IOS 12.0 IP Services Commands

|            |   |
|------------|---|
| <b>log</b> | <p>(Optional) Causes an informational logging message about the packet that matches the entry to be sent to the console. (The level of messages logged to the console is controlled by the <b>logging console</b> command.)</p> <p>The message includes the access list number, whether the packet was permitted or denied; the protocol, whether it was TCP, UDP, ICMP or a number; and, if appropriate, the source and destination addresses and source and destination port numbers. The message is generated for the first packet that matches, and then at 5-minute intervals, including the number of packets permitted or denied in the prior 5-minute interval.</p> |
|------------|---|

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The combination of Catalyst 5000 Switch Family and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 1.11.

The combination of Catalyst 5000 Switch Family and Goldman satisfies this limitation. *See* Ex. A-3, limitation 1.11.

The combination of Catalyst 5000 Switch Family and da Rocha satisfies this limitation. *See* Ex. A-6, limitation 1.11.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2635. Cisco Catalyst 5000 discloses claim 2, “[t]he method of claim 2, wherein the

**access-list (IP extended)**

To define an extended IP access list, use the extended version of the **access-list** global configuration command. To remove the access lists, use the **no** form of this command.

**access-list** *access-list-number* [**dynamic** *dynamic-name* [*timeout minutes*]] {**deny** | **permit**} *protocol source source-wildcard destination destination-wildcard* [**precedence precedence**] [**tos tos**] [**log** | **log-input**]

assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

IOS 12.0 IP Services Commands

IOS 12.0 Quality of Service Commands

3. ***Claim 3***

**rate-limit**

To configure committed access rate (CAR) and Distributed CAR (DCAR) policies, use the **rate-limit** interface configuration command. To remove the rate limit from the configuration, use the **no** form of this command.

**rate-limit** (*input* | *output*) [**access-group** [**rate-limit**] *acl-index*] *bps burst-normal burst-max conform-action action exceed-action action*

**no rate-limit** (*input* | *output*) [**access-group** [**rate-limit**] *acl-index*] *bps burst-normal burst-max conform-action action exceed-action action*

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2636. Cisco Catalyst 5000 discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

IP Access Lists may be configured to filter (or match when used for QoS)

|                 |  |
|-----------------|--|
| <i>protocol</i> | Name or number of an IP protocol. It can be one of the keywords <b>eigrp</b> , <b>gre</b> , <b>icmp</b> , <b>igmp</b> , <b>igrp</b> , <b>ip</b> , <b>ipinip</b> , <b>nos</b> , <b>ospf</b> , <b>tcp</b> , or <b>udp</b> , or an integer in the range 0 to 255 representing an IP protocol number. To match any Internet protocol (including ICMP, TCP, and UDP) use the keyword <b>ip</b> . Some protocols allow further qualifiers described below. |
|-----------------|--|

based on protocol, including OSPF.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

## IOS 12.0 IP Services Commands

- b. *constructing a topological representation of the computer network from the information.*

2637. Cisco Catalyst 5000 discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

IP Access Lists may be configured to filter (or match when used for QoS) based on protocol, including OSPF. The OSPF protocol uses OSPF messages. NetFuel contends OSPF builds a topological representation of the computer network. Under NetFuel’s application of this limitation, Cisco Catalyst 5000 Switch Family satisfies this limitation.

|                 |  |
|-----------------|--|
| <i>protocol</i> | Name or number of an IP protocol. It can be one of the keywords <b>eigrp</b> , <b>gre</b> , <b>icmp</b> , <b>igmp</b> , <b>igrp</b> , <b>ip</b> , <b>ipinip</b> , <b>nos</b> , <b>ospf</b> , <b>tcp</b> , or <b>udp</b> , or an integer in the range 0 to 255 representing an IP protocol number. To match any Internet protocol (including ICMP, TCP, and UDP) use the keyword <b>ip</b> . Some protocols allow further qualifiers described below. |
|-----------------|--|

## IOS 12.0 IP Services Commands

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2638. Cisco Catalyst 5000 discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

IP Access Lists may be configured to filter (or match when used for QoS) based on protocol, including OSPF. The OSPF protocol uses OSPF messages. NetFuel contends OSPF uses a numerical method model using a

|                 |  |
|-----------------|--|
| <i>protocol</i> | Name or number of an IP protocol. It can be one of the keywords <b>eigrp</b> , <b>gre</b> , <b>icmp</b> , <b>igmp</b> , <b>igrp</b> , <b>ip</b> , <b>ipinip</b> , <b>nos</b> , <b>ospf</b> , <b>tcp</b> , or <b>udp</b> , or an integer in the range 0 to 255 representing an IP protocol number. To match any Internet protocol (including ICMP, TCP, and UDP) use the keyword <b>ip</b> . Some protocols allow further qualifiers described below. |
|-----------------|--|

numerical method. Under NetFuel’s application of this limitation, Cisco Catalyst 5000 Switch Family satisfies this limitation.

## IOS 12.0 IP Services Commands

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2639. Cisco Catalyst 5000 discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

IP Access Lists may be configured to filter (or match when used for QoS) based on protocol, including OSPF. The OSPF protocol uses OSPF messages. NetFuel contends OSPF uses the Dijkstra Self Stabilization Algorithm to model. Under NetFuel's application of this limitation, Cisco

|                 |  |
|-----------------|--|
| <i>protocol</i> | Name or number of an IP protocol. It can be one of the keywords <b>eigrp</b> , <b>gre</b> , <b>icmp</b> , <b>igmp</b> , <b>igrp</b> , <b>ip</b> , <b>ipinip</b> , <b>nos</b> , <b>ospf</b> , <b>tcp</b> , or <b>udp</b> , or an integer in the range 0 to 255 representing an IP protocol number. To match any Internet protocol (including ICMP, TCP, and UDP) use the keyword <b>ip</b> . Some protocols allow further qualifiers described below. |
|-----------------|--|

Catalyst 5000 Switch Family satisfies this limitation.

## IOS 12.0 IP Services Commands

### 6. *Claim 7*

#### a. *A computer network, comprising:*

2640. Cisco Catalyst 5000 discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

#### b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

Catalyst 5505 The Catalyst 5505 is a high-performance, 5-slot chassis for the evolving Catalyst 5000 Series. It combines the size of the original Catalyst 5000 with the performance boost and added features of the Catalyst 5500. The Catalyst 5505 is ideal for high-performance wiring closets and data applications. The Catalyst 5505 supports Cisco's Investment Protection standards because it can redeploy all the line cards in an existing Catalyst 5000 series. The Catalyst 5505 can be configured for backbone applications with feature-rich, scalable 100/1000 Ethernet, ATM, and FDDI, as well as optional redundant Supervisor Engines and power supplies. In the wiring closet, switched 10/100 Ethernet, Token Ring, and ATM modules provide high-performance connectivity.

Cisco 5000 Family – Multilayer Switches”, June 2000 at 2<sup>15</sup>; *see also* Cisco 5000 Family – Multilayer Switches.

2641. Cisco Catalyst 5000 discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

<sup>15</sup> (<https://www.andovercg.com/datasheets/cisco-5000-5500-series-switches.pdf>)



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

2642. Cisco Catalyst 5000 discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

2643. Cisco Catalyst 5000 discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2644. Cisco Catalyst 5000 discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2645. Cisco Catalyst 5000 discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2646. Cisco Catalyst 5000 discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

## **Release Notes for Catalyst 5000 Family RSM/VIP2 Cisco IOS 12.0 Software Releases**

January 2, 2001

Current Release:

12.0(15)

Hardware Supported

]

The following switches are supported in Release 12.0:

- Catalyst 5000
- Catalyst 5002
- Catalyst 5500

\_\_\_\_\_

dates

(For detailed descriptions of the new hardware features for Release 12.0, refer to the *Cross-Platform Release Notes for Cisco IOS port, Release 12.0*.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Catalyst 5K Family RSM/VIP2 Cisco IOS 12.0 Release Notes; *see also* RSM, RSFC, Cisco IOS Release 12.0W5 Release Notes

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2647. Cisco Catalyst 5000 discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2648. Cisco Catalyst 5000 discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2649. Cisco Catalyst 5000 discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the optimal policy;*

2650. Cisco Catalyst 5000 discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2651. Cisco Catalyst 5000 discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2652. Cisco Catalyst 5000 discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2653. Cisco Catalyst 5000 discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

Catalyst 5505 The Catalyst 5505 is a high-performance, 5-slot chassis for the evolving Catalyst 5500 Series. It combines the size of the original Catalyst 5000 with the performance boost and added features of the Catalyst 5500. The Catalyst 5505 is ideal for high-performance wiring closets and data applications. The Catalyst 5505 supports Cisco’s Investment Protection standards because it can redeploy all the line cards in an existing

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Catalyst 5000 series. The Catalyst 5505 can be configured for backbone applications with feature-rich, scalable 100/1000 Ethernet, ATM, and FDDI, as well as optional redundant Supervisor Engines and power supplies. In the wiring closet, switched 10/100 Ethernet, Token Ring, and ATM modules provide high-performance connectivity.

Cisco 5000 Family – Multilayer Switches”, June 2000 at 2<sup>16</sup>; *see also* IOS 12.0 Configuring IP Services; IOS 12.0 Quality of Service Commands

ACL/QoS Agents communicate with the command line interface and supporting software, which are also software agents under NetFuel’s interpretation of the term.

To the extent this limitation is not met, it is rendered obvious by Kasteleijn.

*See* Ex. A-2, limitation 11.

**9. Claim 12**

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2654. Cisco Catalyst 5000 discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

The ACL/QoS software may be distributed among different line cards and controlled by a supervisor module in the Catalyst 5000 Switch Family. Under NetFuel’s interpretation of this claim, one ACL/QoS software agent may “communicate directly with another” ACL/QoS.

Catalyst 5505 The Catalyst 5505 is a high-performance, 5-slot chassis for the evolving Catalyst 5500 Series. It combines the size of the original Catalyst 5000 with the performance boost and added features of the Catalyst 5500. The Catalyst 5505 is ideal for high-performance wiring closets and data applications. The Catalyst 5505 supports Cisco’s Investment Protection standards because it can redeploy all the line cards in an existing Catalyst 5000 series. The Catalyst 5505 can be configured for backbone applications with feature-rich, scalable 100/1000 Ethernet, ATM, and FDDI, as well as optional redundant Supervisor Engines and power supplies. In the wiring closet, switched 10/100 Ethernet, Token Ring, and ATM modules provide high-performance connectivity.

“Cisco 5000 Family – Multilayer Switches”, June 2000 at 2<sup>17</sup>; *see also* IOS 12.0 Configuring IP Services; IOS 12.0 Quality of Service Commands

To the extent this limitation is not met, it is rendered obvious by Kasteleijn.

---

<sup>16</sup> (<https://www.andovercg.com/datasheets/cisco-5000-5500-series-switches.pdf>)

<sup>17</sup> (<https://www.andovercg.com/datasheets/cisco-5000-5500-series-switches.pdf>)

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

See Ex. A-2, limitation 12.

10. **Claim 16**

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2655. Cisco Catalyst 5000 discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:


**Release Notes for  
Catalyst 5000 Family RSM/VIP2 Cisco IOS 12.0 Software  
Releases**

January 2, 2001

Current Release:  
12.0(15)

Previous Releases:  
12.0(14), 12.0(13), 12.0(12), 12.0(11), 12.0(10), 12.0(9), 12.0(8), 12.0(7), 12.0(6), 12.0(5), 12.0(4), 12.0(3), 12.0(2), 12.0(1)

---

 **Note** You can find the most current Cisco IOS documentation on Cisco.com. These electronic documents may contain updates and modifications made after the hard copy documents were printed.

---

These release notes for the Catalyst 5000 family Route Switch Module (RSM) and the optional Versatile Interface Processor 2 (VIP2) support Cisco IOS Release 12.0. These release notes are updated to describe new memory requirements, hardware support, software platform deferrals, and changes to the microcode or modem code and related documents.

**Hardware Supported**

The following switches are supported in Release 12.0:

- Catalyst 5000
- Catalyst 5002
- Catalyst 5500

For detailed descriptions of the new hardware features for Release 12.0, refer to the *Cross-Platform Release Notes for Cisco IOS Release 12.0*.

Catalyst 5K Family RSM/VIP2 Cisco IOS 12.0 Release Notes; *see also* RSM, RSFC, Cisco IOS Release 12.0W5 Release Notes

11. **Claim 17**

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

2656. Cisco Catalyst 5000 discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

IOS controls the operation of the ACL and QoS agents based at least on CLI commands and configuration files, which are predetermined criteria under NetFuel’s application of the claims.

To the extent this limitation is not met, it is rendered obvious by Kasteleijn. See Ex. A-2, limitation 17.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2657. Cisco Catalyst 5000 discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

Cisco IOS instantiates the ACL or QoS software and thus “spawns” it under NetFuel’s application of the claims.

To the extent this limitation is not met, it is rendered obvious by Kasteleijn. See Ex. A-2, limitation 18.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2658. Cisco Catalyst 5000 discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

NetFuel contends this limitation is inherent in IOS. Under NetFuel’s application of the claims, this limitation is anticipated or rendered obvious by Cisco IOS.

To the extent this limitation is not met, it is rendered obvious by Kasteleijn. See Ex. A-2, limitation 19.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*database to store the policy for the agent.*

Cisco Catalyst 5000 discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

To create a standard access list, use one of the following commands in global configuration mode:

| Command  | Purpose   |
|--|---|
| <code>access-list access-list-number {deny   permit} source [source-wildcard] [log]</code> | Define a standard IP access list using a source address and wildcard.   |
| <code>access-list access-list-number {deny   permit} any [log]</code>                      | Define a standard IP access list using an abbreviation for the source and source mask of 0.0.0.0 255.255.255.255. |

IOS 12.0 Configuring IP Services; *see also* IOS 12.0 Quality of Service Commands.

IOS stores the configuration for ACLs and QoS in a policy database under NetFuel’s application of the claims.

To the extent this limitation is not met, it is rendered obvious by Kasteleijn.

*See* Ex. A-2, limitation 21.

### 15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2659. Cisco Catalyst 5000 discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

---

|                    |   |
|--------------------|---|
| Network Management | CiscoWorks for Switched Internetworks GUI management including the following: |
|                    | CiscoView   |
|                    | VlanDirector  |
|                    | Traffic Director  |
|                    | User Tracking   |
|                    | ATMDirector   |

“Cisco 5000 Family – Multilayer Switches”, June 2000 at 6  
<https://www.andovercg.com/datasheets/cisco-5000-5500-series-switches.pdf>

To the extent this limitation is not met, it is rendered obvious by Kasteleijn. *See* Ex. A-2, limitation 22.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Additionally, the combination of Catalyst 5000 Switch Family and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 22.0.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2660. Cisco Catalyst 5000 discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2661. Cisco Catalyst 5000 discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The combination of Catalyst 5000 Switch Family and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 26.0.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2662. Cisco Catalyst 5000 discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

The combination of Catalyst 5000 Switch Family and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 29.0.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform a method comprising*

2663. Cisco Catalyst 5000 discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

“Unique Traffic Management Capabilities

Embedded traffic management functionality is a critical component of the Catalyst 5000 design. The Catalyst 5000 supports extensive input buffers across its switching modules that function with buffer management software to eliminate packet loss during peak traffic periods.

The Catalyst 5000 architecture supports multiple levels of data prioritization, with each interface separately user-configurable as high or low priority. Maintenance of separate logical queues for each priority class reduces buffering delays, a critical issue with typically bursty network traffic and delay-sensitive traffic types such as voice and video.

The Catalyst 5000 family supports switched port analyzer software. This software gives network managers the ability to track traffic on any one of the 5000's switching ports.

Cisco is the only vendor to support embedded remote monitoring (RMON) across all of its switching platforms. Because RMON is fully integrated with the CiscoView device management application, network managers can view, configure and set alarms on a graphical representation of the switch.

Catalyst 5000 also supports RFC 1271 standard-compliant RMON across all ports, enabling users to centrally manage, administer and control traffic monitoring for all switched LAN segments simultaneously. Offering centralized control that greatly reduces administrative overhead, the embedded RMON agent can save the user up to 80 percent of the cost of using specialized hardware probes on each LAN segment.

“Cisco Delivers Switching for Wiring Closets” March 28, 1995.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2664. Cisco Catalyst 5000 discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

2665. Cisco Catalyst 5000 discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

d. *is capable of perceiving its own state; and*

2666. Cisco Catalyst 5000 discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself;*

2667. Cisco Catalyst 5000 discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2668. Cisco Catalyst 5000 discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2669. Cisco Catalyst 5000 discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2670. Cisco Catalyst 5000 discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

2671. Cisco Catalyst 5000 discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2672. Cisco Catalyst 5000 discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2673. Cisco Catalyst 5000 discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2674. Cisco Catalyst 5000 discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2675. Cisco Catalyst 5000 discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2676. Cisco Catalyst 5000 discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

**22. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2677. Cisco Catalyst 5000 discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2678. Cisco Catalyst 5000 discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

**23. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2679. Cisco Catalyst 5000 discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2680. Cisco Catalyst 5000 discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

**T. Anticipation by and/or Obviousness in View of** [REDACTED]

2681. As explained in detail below and in the chart attached as Ex. A-10, [REDACTED] anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2682. [REDACTED] discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” '730 Pat. at Cl. 1. For example:

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED] 00000014 at PDF p.2; see also CSI-NF-00000005 at 6, 8.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2683. [REDACTED] discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:



CSI-NF-00000014 at PDF p.3; *see also* CSI-NF-00000005 at 7, 9, 10.

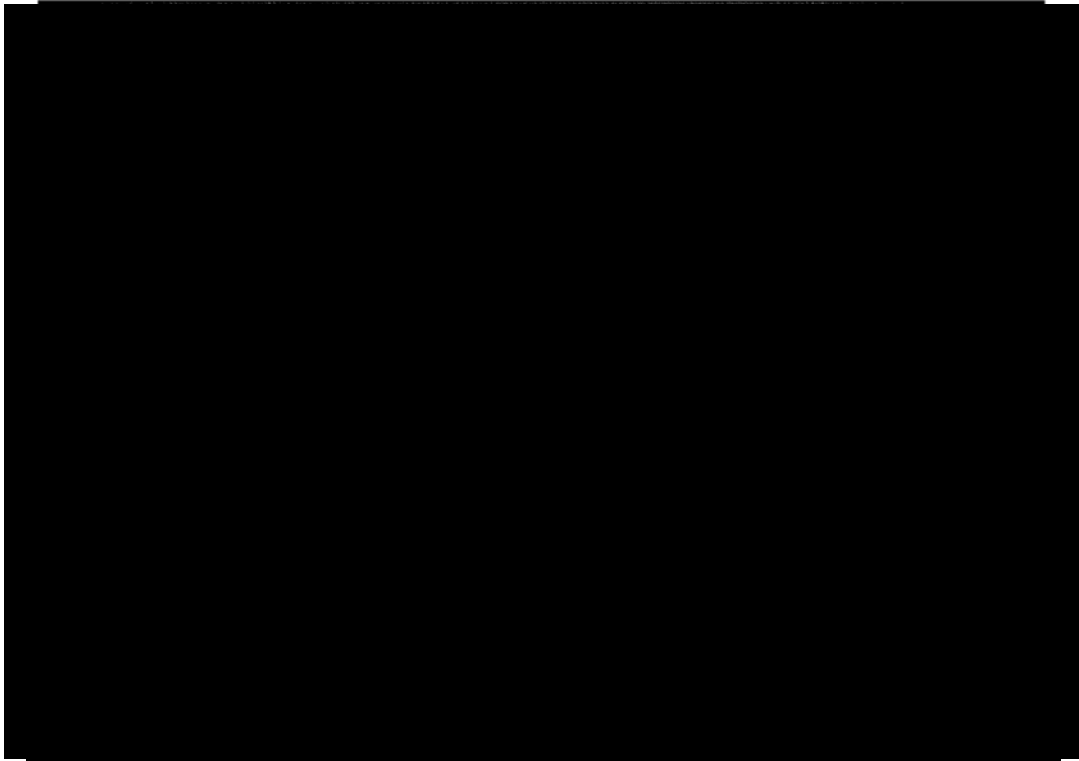
To the extent this limitation is not disclosed by [REDACTED] it is rendered obvious in combination with Kasteleijn. *See Ex. A-2, limitation 1.1.* It would have been obvious to implement [REDACTED] as a software agent to obtain the benefits of the MCs in Kasteleijn, for example, to maintain state during a resource crash or migration, and to improve resource reusage and reduce resource consumption.

To the extent this limitation is not disclosed by [REDACTED] it is rendered obvious in combination with Pandya. *See Ex. A-4, limitation 1.1.* It would have been obvious to implement [REDACTED] as agents in communication with a control unit as taught by Pandya, in order to, for example, permit the [REDACTED] software to be managed by an external entity or entities.

- c. *is able to communicate with other software agents in the computer network;*

2684. [REDACTED] discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-00000005 at 7

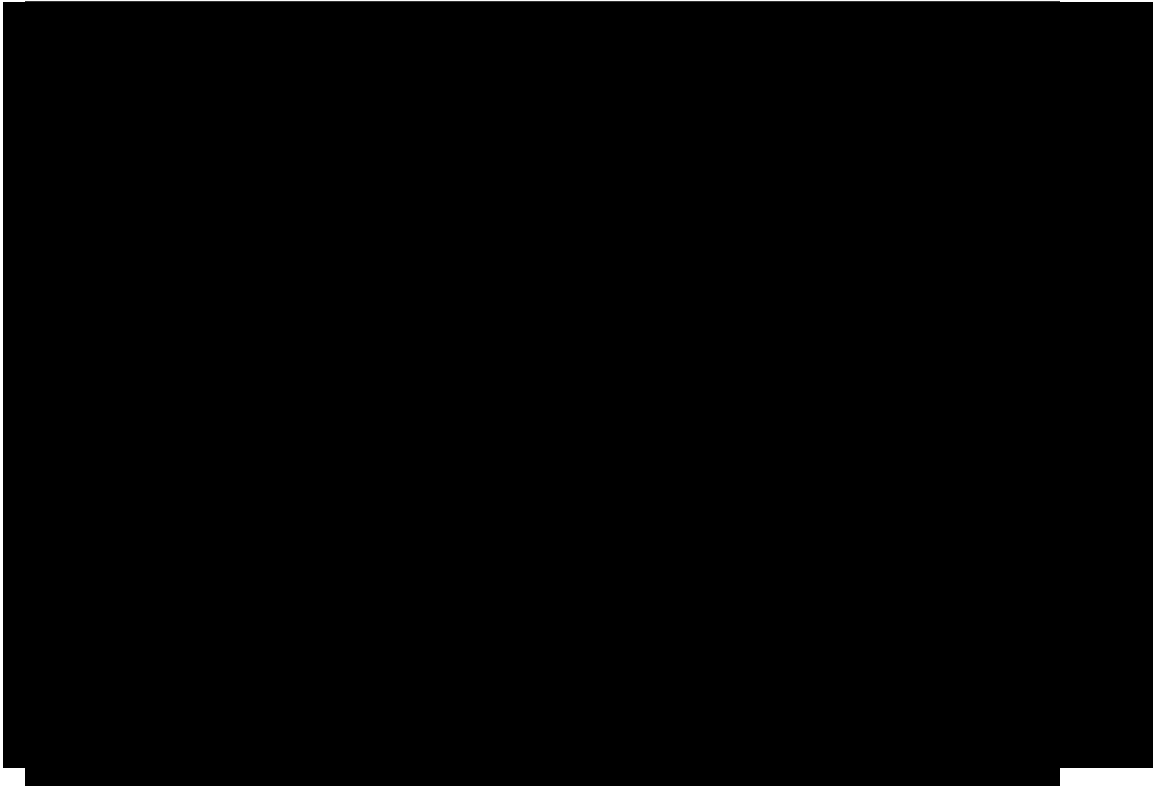
To the extent this limitation is not disclosed by [REDACTED], it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.2.

To the extent this limitation is not disclosed by [REDACTED] it is rendered obvious in combination with Pandya. *See* Ex. A-4, limitation 1.2.

d. *is capable of perceiving its own state*

2685. [REDACTED] this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-00000005 at 7; *see also* CSI-NF-00000014 at PDF p.4

To the extent this limitation is not disclosed by [REDACTED]  
[REDACTED] it is rendered obvious in combination with Kasteleijn. See Ex.  
A-2, limitation 1.3.

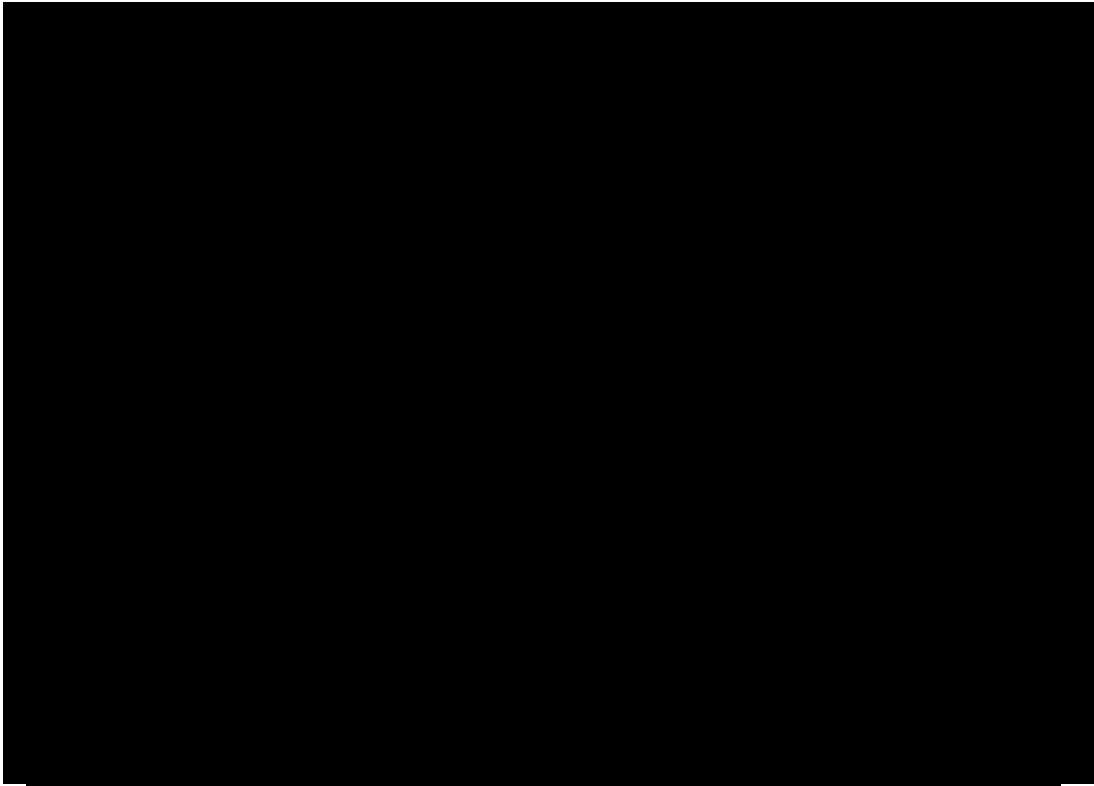
To the extent this limitation is not disclosed by [REDACTED]  
[REDACTED] it is rendered obvious in combination with Pandya.

See Ex. A-4, limitation 1.3.

e. *and is able to clone itself,*

2686. [REDACTED] discloses this element of claim 1, “and is able  
to clone itself.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-00000005 at 7; *see also* CSI-NF-00000014 at PDF p.4

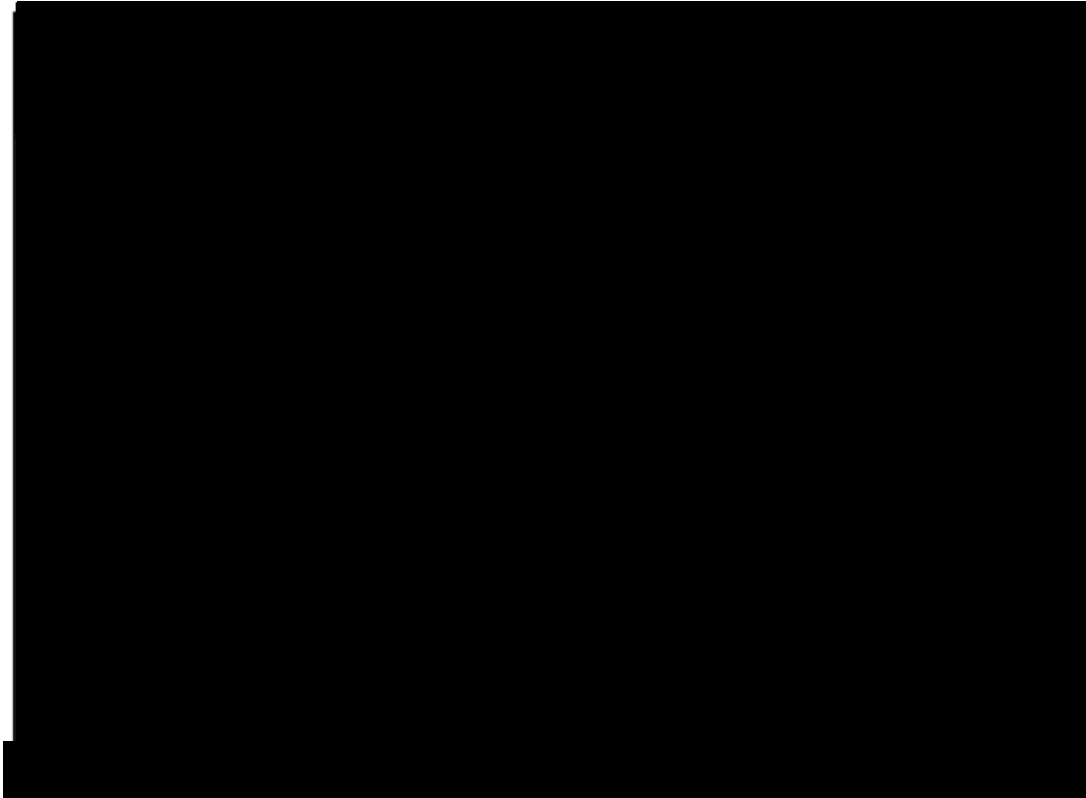
To the extent this limitation is not disclosed by [REDACTED]  
[REDACTED] it is rendered obvious in combination with Kasteleijn. See Ex. A-2, limitation 1.4.

To the extent this limitation is not disclosed by [REDACTED]  
[REDACTED] it is rendered obvious in combination with Pandya. See Ex. A-4, limitation 1.4.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2687. [REDACTED] discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-00000005 at 9; *see also id.* at 10; CSI-NF-00000014 at PDF p.3

To the extent this limitation is not disclosed by [REDACTED]  
[REDACTED] it is rendered obvious in combination with Kasteleijn. See Ex. A-2, limitation 1.5.

To the extent this limitation is not disclosed by [REDACTED]  
[REDACTED] it is rendered obvious in combination with Pandya. See Ex. A-4, limitation 1.5.

g. *monitoring the computer network;*

2688. [REDACTED] discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

[REDACTED]

[REDACTED]

[REDACTED]



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Since DoS attacks at Cisco routers are targeted at local addresses, we use

[REDACTED]

[REDACTED]

[REDACTED]

CSI-NF-00000014 at PDF p.4; *see also* CSI-NF-00000005 at 9

To the extent this limitation is not disclosed by [REDACTED], it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.7.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2689. [REDACTED] discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

[REDACTED]

[REDACTED]

[REDACTED]

CSI-NF-00000014 at PDF p.4; *see also* CSI-NF-00000005 at 9

Additionally, [REDACTED] discloses this limitation under NetFuel's interpretation of this term. The initial rate limiting access group is the test policy. Execution and enforcement of these test policies then models and reflects the actual behavior of the computer network, which can then be monitored by a user or other software that can then instantiate a revised optimal policy.

2690. To the extent this limitation is not met, it would have been obvious in view of da Rocha. See Ex. A-6, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. faults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings for [REDACTED]), as taught by da Rocha.

2691. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine [REDACTED] and Hellerstein to implement these models.

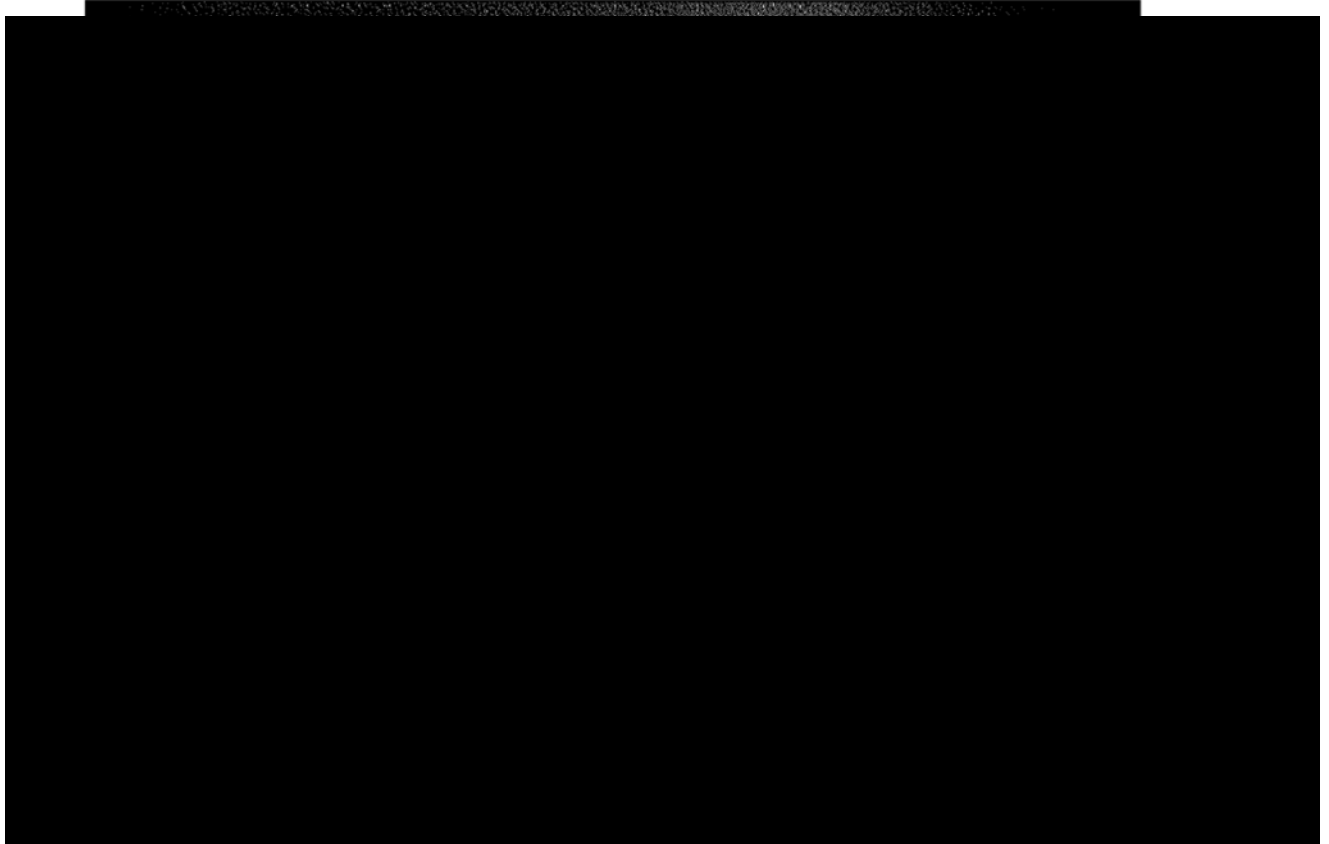
- i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction algorithm*

2692. [REDACTED] discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1.

For example:



CSI-NF-00000006 at 6

2693. To the extent this limitation is not met, it would have been obvious in view of da Rocha. See Ex. A-6, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. faults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings for [REDACTED] as taught by da Rocha.

2694. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine [REDACTED] and Hellerstein to implement these models.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2695. [REDACTED] discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl.

1. For example:

2696. Additionally, [REDACTED] discloses this limitation under NetFuel’s interpretation of this term. The initial rate limiting access group is the test policy. Execution and enforcement of these test policies then models and reflects the actual behavior of the computer network, which can then be monitored by a user or other software that can then instantiate a revised optimal policy.

2697. To the extent this limitation is not met, it would have been obvious in view of da Rocha. See Ex. A-6, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. faults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings for [REDACTED], as taught by da Rocha.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2698. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine [REDACTED] and Hellerstein to implement these models.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2699. [REDACTED] discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

2700. Additionally, [REDACTED] discloses this limitation under NetFuel’s interpretation of this term. The initial rate limiting access group is the test policy. Execution and enforcement of these test policies then models and reflects the actual behavior of the computer network, which can then be monitored by a user or other software that can then instantiate a revised optimal policy.

2701. To the extent this limitation is not met, it would have been obvious in view of da Rocha. See Ex. A-6, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. faults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings for [REDACTED]), as taught by da Rocha.

2702. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine [REDACTED] and Hellerstein to implement these models.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2703. [REDACTED] discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of da Rocha.

See Ex. A-6, limitation 1.11.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2704. [REDACTED] discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

CSI-NF-00000014 at PDF p.3; *see also* CSI-NF-00000005 at 7, 9, 10.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the predefined task; and*

2705. [REDACTED] discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

2706. Furthermore, it would have been obvious to combine [REDACTED] with the OSFP routing protocol (RFC 2328). Under NetFuel’s interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. [REDACTED]

[REDACTED] OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement [REDACTED] system on network devices using OSPF. *See also* CSI-NF-00000014 at PDF p.2

b. *constructing a topological representation of the computer network from the information.*

2707. [REDACTED] discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

2708. Furthermore, it would have been obvious to combine [REDACTED] with the OSFP routing protocol (RFC 2328). Under NetFuel’s interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. [REDACTED]

[REDACTED] OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement [REDACTED] system on network devices using OSPF. *See also* CSI-NF-00000014 at PDF p.2.

4. ***Claim 4***

a. *The method of claim 1, wherein the modeling uses a numerical*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*method.*

2709. [REDACTED] discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

2710. Furthermore, it would have been obvious to combine [REDACTED]  
[REDACTED] with the OSPF routing protocol (RFC 2328). [REDACTED]  
[REDACTED]  
[REDACTED] OSPF routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement [REDACTED] system on network devices using OSPF. *See also* CSI-NF-00000014 at PDF p.2.

**5. Claim 6**

a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2711. [REDACTED] discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

2712. Furthermore, it would have been obvious to combine [REDACTED]  
[REDACTED] with the OSPF routing protocol (RFC 2328). [REDACTED]  
[REDACTED]  
[REDACTED] routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement [REDACTED] system on network devices using OSPF. *See also* CSI-NF-00000014 at PDF p.2.

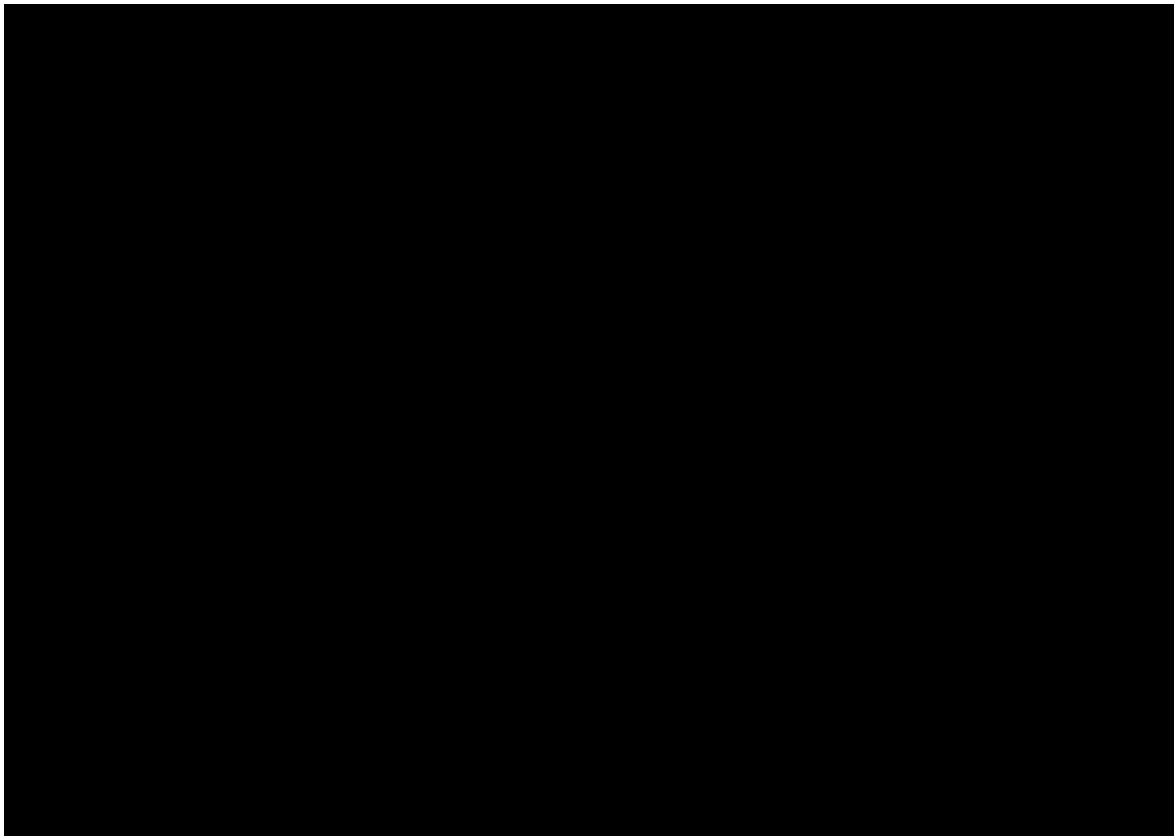
**6. Claim 7**

a. *A computer network, comprising:*

2713. [REDACTED] discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-00000006 at 6

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2714. [REDACTED] discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

2715. [REDACTED] discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See also* CSI-NF-00000005 at 11.

- d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

2716. [REDACTED] discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2717. [REDACTED] discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2718. [REDACTED] discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

2719. [REDACTED] discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

[REDACTED]

CSI-NF-00000005 at 11; *see also id.* at 8.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2720. [REDACTED] discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*of a network component*

2721. [REDACTED] discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the prediction;*

2722. [REDACTED] discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2723. [REDACTED] discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2724. [REDACTED] discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

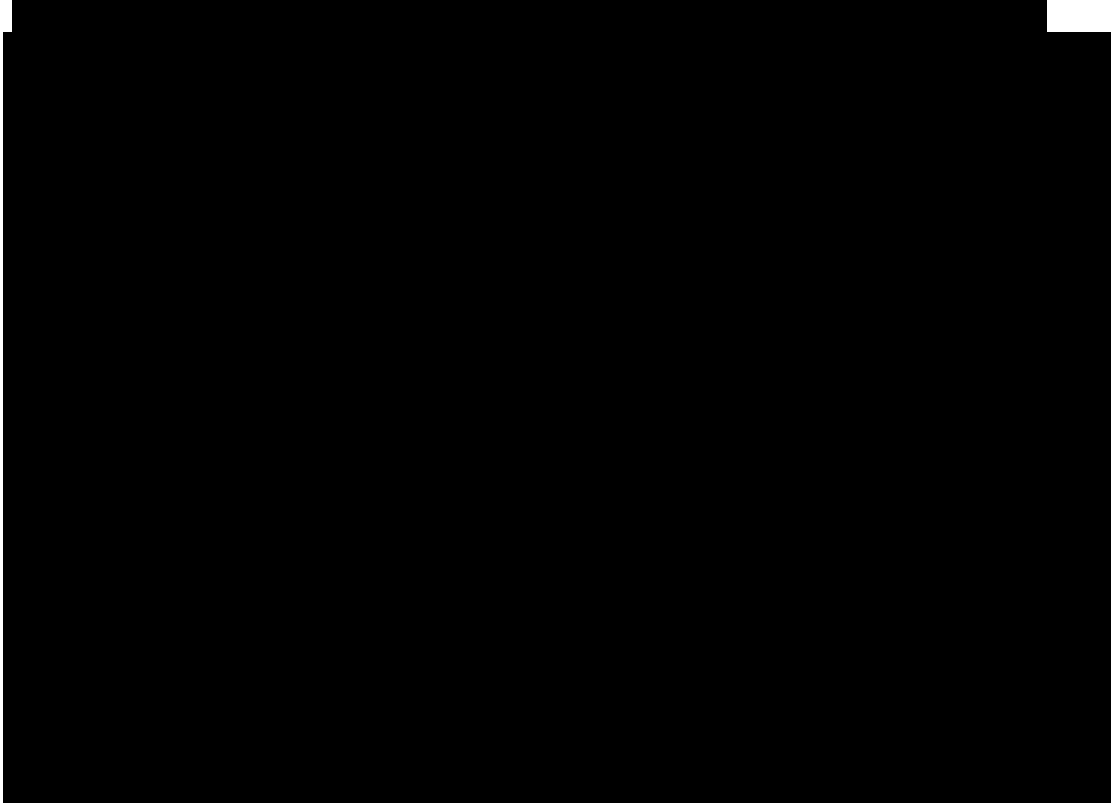
2725. [REDACTED] discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2726. [REDACTED] discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:



CSI-NF-00000005 at 7.

2727. To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 11.0.

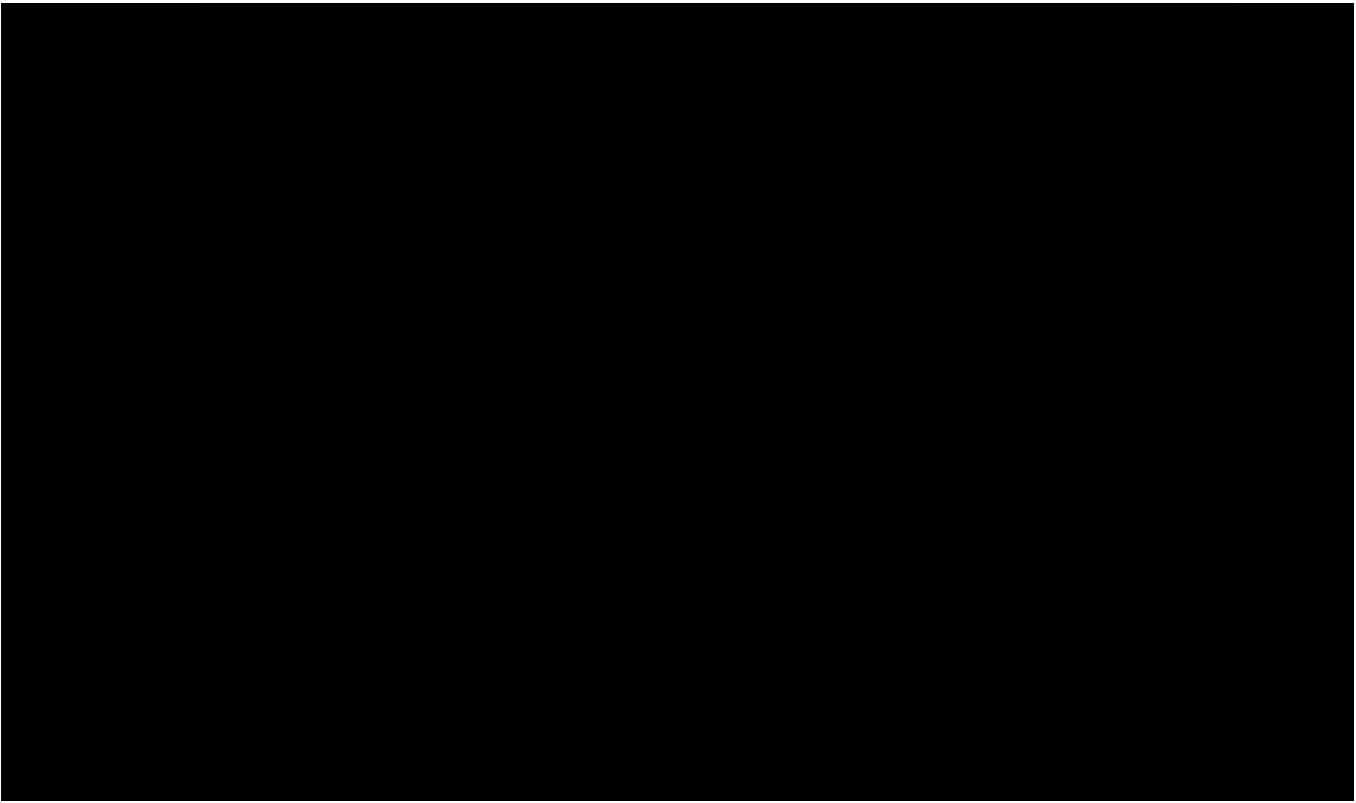
9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism comprises a secure communications protocol.*

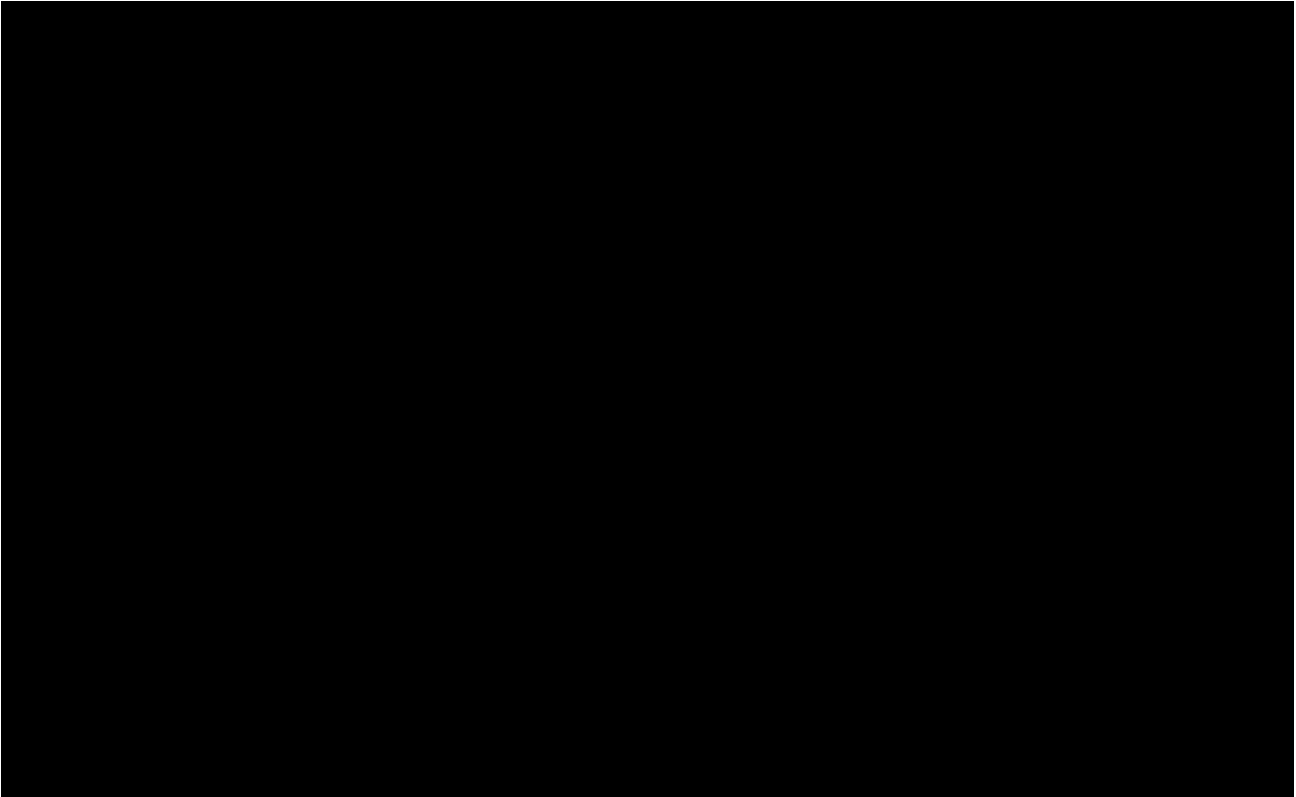
2728. [REDACTED] discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:



CSI-NF-00000005 at 7.

2729. To the extent [REDACTED] not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 11.0.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



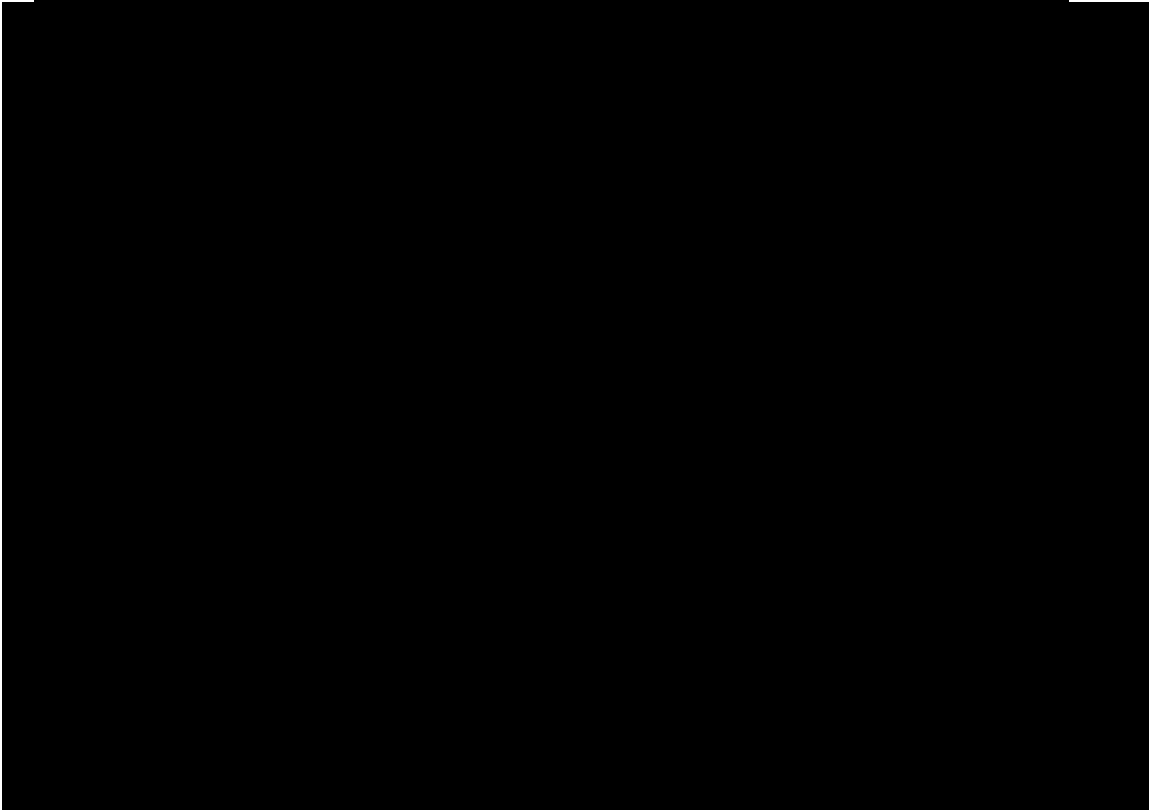
CSI-NF-00000005 at 7.

2730. Claim 16

- b. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2731. [REDACTED] discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**



CSI-NF-00000005 at 7; *see also id.* at 8, 11.

2732. To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 12.0.

10. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2733. [REDACTED] discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

2734. To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 17.0.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2735. [REDACTED] discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 18.0.

12. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2736. [REDACTED] discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

2737. NetFuel contends this limitation is inherent and thus disclosed by [REDACTED]  
[REDACTED]

2738. To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 19.0.

13. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2739. [REDACTED] discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

[REDACTED]

[REDACTED]

[REDACTED]

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

[REDACTED]

[REDACTED]

[REDACTED]

CSI-NF-00000014 at PDF p.3; *see also* CSI-NF-00000005 at 7, 9, 10.

To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya.

*See* Exs. A-2, A-4 at limitation 21.0.

14. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2740. [REDACTED] discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

2741. To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 22.0.

15. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2742. [REDACTED] claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

16. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

2743. [REDACTED] discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

2744. Additionally, [REDACTED] discloses this limitation under NetFuel’s interpretation of this term. The initial rate limiting access group is the test policy. Execution and enforcement of these test policies then models and reflects the actual behavior of the computer network, which can then be monitored by a user or other software that can then instantiate a revised optimal policy.

2745. To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 26.0.

17. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2746. [REDACTED] claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

CSI-NF-00000014 at PDF p.3.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2747. Additionally, [REDACTED] discloses this limitation under NetFuel's interpretation of this term. The initial rate limiting access group is the test policy. Execution and enforcement of these test policies then models and reflects the actual behavior of the computer network, which can then be monitored by a user or other software that can then instantiate a revised optimal policy.

2748. To the extent [REDACTED] does not expressly or implicitly disclose this limitation, it would have been obvious in view of Kasteleijn or Pandya. *See* Exs. A-2, A-4 at limitation 29.0.

18. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2749. [REDACTED] discloses the preamble of claim 30, "[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising." '730 Pat. at Cl. 30. For example:

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

CSI-NF-00000014 at PDF p.2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2750. [REDACTED] discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2751. [REDACTED] discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2752. [REDACTED] discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2753. [REDACTED] discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2754. [REDACTED] discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2755. [REDACTED] this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the computer network*

2756. [REDACTED] discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a predictive algorithm;*

2757. [REDACTED] discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2758. [REDACTED] discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2759. [REDACTED] discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

19. ***Claim 31***

a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2760. [REDACTED] discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

20. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2761. [REDACTED] discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2762. [REDACTED] this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

21. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2763. [REDACTED] discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2764. [REDACTED] discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

22. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*determining uses a numerical method.*

2765. [REDACTED] discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

23. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2766. [REDACTED] this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

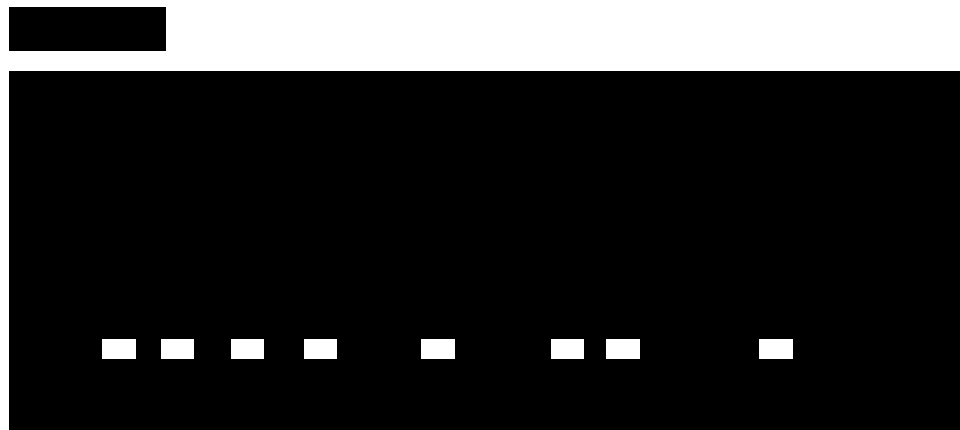
**U. Anticipation by and/or Obviousness in View of Cisco Receive ACL (“rACL”), marketed by Cisco by April 22, 2002 at the latest.**

2767. As explained in detail below and in the chart attached as Ex. A-11, Cisco rACL anticipates and renders obvious the claims of the ’730 Patent at least under the apparent application of the claims in NetFuel’s infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2768. Under NetFuel’s Cisco rACL discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:





**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED] NF-00000013 at 4; *see also id.* at 12; CSI-NF-00000018 at 2; CSI-NF-00000025.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2769. Cisco rACL discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

[REDACTED]

[REDACTED]

CSI-NF-00000026; *see also* CSI-NF-00000032; CSI-NF-00000033; GSR: Receive Access Control Lists<sup>18</sup> at 3.

rACL software is an agent under NetFuel’s application of the claim. To the extent rACL fails to explicitly or implicitly disclose a “software agent”, it would have been obvious to in view of Kasteleijn. *See* Ex. A-2, limitation 1.1. It would have been obvious to implement rACL software as a software agent to obtain the benefits of the MCs in Kasteleijn, for example, to maintain state during a resource crash or migration, and to improve resource reuse and reduce resource consumption.

- c. *is able to communicate with other software agents in the computer network;*

2770. Cisco rACL discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

[REDACTED]

■ GSR: Receive Access Control Lists, Cisco Systems, Inc. <https://www.cisco.com/c/en/us/support/docs/ip/access-lists/43861-racl.html> (last updated Feb. 23, 2006).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED] 00000020; *see also* GSR: Receive Access Control Lists at 3.

2771. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 1.2.

d. *is capable of perceiving its own state*

2772. Cisco rACL discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

The LC performing the actual filtering (in other words, the LC receiving the traffic filtered by the rACL) will have increased CPU utilization because of the processing of the rACL. This increased CPU utilization, however, is caused by a high volume of traffic destined to the GRP; the benefit of the GRP of the rACL protection far outweighs the increased CPU utilization on an LC. The CPU utilization on an LC will vary based on LC engine type. For instance, given the same attack, an engine 3 LC will have lower CPU utilization than an engine 0 LC.

GSR: Receive Access Control Lists at 3.

By protecting the GRP, the rACL helps ensure router and, ultimately, network stability during an attack. As described above, the rACL is processed on the LC CPU, so the CPU utilization on each LC will increase when a large volume of data is directed at the router. On E0/E1 and some E2 bundles, CPU utilization of 100+% might lead to routing protocol and link-layer drops. These drops are localized to the card, and the GRP routing processes are protected, thus maintaining stability. E2 cards with throttling-enabled microcode 5 activate throttling mode when under heavy load and only forward precedence 6 and 7 traffic to the routing protocol. Other engine types have multi-queue architectures; for instance, E3 cards have three queues to the CPU, with routing protocol packets (precedence 6/7) in a separate, high-priority queue. High LC CPU, unless high-precedence packets cause it, will not result in routing protocol drops. Packets to the lower priority queues will be tail-dropped. Finally, E4-based cards have eight queues to the CPU, with one dedicated to routing protocol packets.

GSR: Receive Access Control Lists at 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2773. Under NetFuel's interpretation, each rACL agent "is at least aware of the state of its own execution with respect to IOS software process or subsystem activity.

2774. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. See Ex. A-2, limitation 1.3.

e. *and is able to clone itself,*

2775. Cisco rACL discloses this element of claim 1, "and is able to clone itself." '730 Pat. at Cl. 1. For example:

[REDACTED]

CSI-NF-00000020.

Traffic that enters an LC is first sent to the local CPU of the LC, and packets that require processing by the GRP are queued for forwarding to the route processor. The receive ACL is created on the GRP and then pushed down to the CPUs of the various LCs. Before traffic is sent from the LC CPU to the GRP, the traffic is compared to the rACL. If permitted, the traffic passes to the GRP, while all other traffic is denied. The rACL is inspected prior to the LC to GRP rate-limiting function. Since the rACL is used for all receive adjacencies, some packets that are handled by the LC CPU (such as echo requests) are subject to rACL filtering as well. This needs to be taken into account when designing rACL entries.

GSR: Receive Access Control Lists at 3; *see also* CSI-NF-00000023 at PDF p.2.

2776. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. See Ex. A-2, limitation 1.4.

f. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent; and*

2777. Cisco rACL discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

**Syntax**

A receive ACL is applied with the following global configuration command to distribute the rACL to each LC in the router.

```
[no] ip receive access-list <num>
```

In this syntax, <num> is defined as follows.

```
<1-199> IP access list (standard or extended)
```

```
<1300-2699> IP expanded access list (standard or extended)
```

**GSR: Receive Access Control Lists at 4**

To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 1.5.

g. *monitoring the computer network;*

2778. Cisco rACL discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

Filtering via a rACL “monitors the network” under NetFuel’s application of the claims.

Additionally, logging filtered packets comprises “monitoring the computer network” under NetFuel’s application of the claims.

Identify protocols used in the network with a classification ACL.

Deploy an rACL that permits all the known protocols that access the GRP. This “discovery” rACL should have both source and destination addresses set to any. Logging can be used to develop a list of source addresses that match the protocol permit statements. In addition to the protocol permit statement, a permit any any log line at the end of the rACL can be used to identify other protocols that would be filtered by the rACL and that might require access to the GRP.

The objective is to determine what protocols the specific network uses. Logging should be used for analysis to determine “what else” might be communicating with the router.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Note: Although the log keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses this keyword might result in an overwhelming number of log entries and possibly high router CPU usage. Use the log keyword for short periods of time and only when needed to help classify traffic.

Review identified packets and begin to filter access to the GRP.

Once the packets filtered by the rACL in step 1 have been identified and reviewed, deploy an rACL with a permit any any statement for the allowed protocols. Just as in step 1, the log keyword can provide more information about the packets that match the permit entries. Using deny any any log at the end can help identify any unexpected packets destined to the GRP. This rACL will provide basic protection and will allow network engineers to ensure that all required traffic is permitted.

The objective is to test the range of protocols that need to communicate with the router without having the explicit range of IP source and destination addresses.

GSR: Receive Access Control Lists at 7–8; *see also* CSI-NF-00000023 at PDF p.2.

2779. The combination of Cisco Receive ACL and NetRanger satisfies this limitation.

See Ex. A-8, limitation 1.11.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2780. Cisco rACL discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example, applying rACLs to interfaces satisfies this limitation under NetFuel’s apparent application of the claims.

2781. To the extent this limitation is not explicitly or implicitly disclosed by Cisco Receive ACL, it would have been obvious to combine with NetRanger. A person of ordinary skill in the art would be motivated (as discussed in the NetRanger manual) to use Cisco devices with rACLs as packet filtering devices monitored by NetRanger sensors for intrusion detection. *See*, Ex. A-8, limitation 1.7. In the combined system, rACL rules and policies, which are designed to filter, block, or otherwise rate-limit traffic, are test policies. Each test policy has values that are set, for example, for permitting or denying traffic and are set by default or modified or configured by a user. The test policies indicate how the traffic in the network should look. For example,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

particular traffic should not be permitted at all on certain ports, or the flow rates for a particular packet types should be at or below a specific threshold.

2782. Execution and enforcement of rACL test policies then models and reflects the actual behavior of the traffic flow on the computer network which can be monitored by NetRanger to determine and then instantiate a revised, optimal policy based on certain triggering events.

2783. For example, when a policy violation is detected by NetRanger, the Director determines the specifics of a particular event that is triggered to select a specific policy, which may include shunning the traffic by reconfiguring the rACLs to deny that traffic.

2784. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. *See* Ex. A-2, limitations 1.7-1.10.

- i. *including predicting a failure of a network component based on a prediction algorithm*

2785. Cisco rACL discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example, under NetFuel’s apparent application of the claims, the combination of Cisco Receive ACL and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 1.8.

2786. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. *See* Ex. A-2, limitations 1.7-1.10.

- j. *wherein said modeling comprises determining appropriate policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

2787. Cisco rACL discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

2788. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 1.9.

2789. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. *See* Ex. A-2, limitations 1.7-1.10.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2790. Cisco rACL discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

2791. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 1.10. As discussed above, the NetRanger system determines an appropriate action for a policy violation which may include reconfiguring the Cisco Receive ACL’s settings.

2792. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. *See* Ex. A-2, limitations 1.7-1.10.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

2793. Cisco rACL discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

“Identify protocols used in the network with a classification ACL.

Deploy an rACL that permits all the known protocols that access the GRP. This “discovery” rACL should have both source and destination addresses set to any. Logging can be used to develop a list of source addresses that match the protocol permit statements. In addition to the protocol permit statement, a permit any any log line at the end of the rACL can be used to identify other protocols that would be filtered by the rACL and that might require access to the GRP.

The objective is to determine what protocols the specific network uses. Logging should be used for analysis to determine “what else” might be communicating with the router.

Note: Although the log keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses this keyword might result in an overwhelming number of log entries and possibly high router CPU usage. Use the log keyword for short periods of time and only when needed to help classify traffic.

Review identified packets and begin to filter access to the GRP.

Once the packets filtered by the rACL in step 1 have been identified and reviewed, deploy an rACL with a permit any any statement for the allowed protocols. Just as in step 1, the log keyword can provide more information about the packets that match the permit entries. Using deny any any log at the end can help identify any unexpected packets destined to the GRP. This rACL will provide basic protection and will allow network engineers to ensure that all required traffic is permitted.

The objective is to test the range of protocols that need to communicate with the router without having the explicit range of IP source and destination addresses.

GSR: Receive Access Control Lists at 7–8; *see also* CSI-NF-00000023 at PDF p.1.

2794. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. See Ex. A-8, limitation 1.11.

**2. Claim 2**

a. *The method of claim 2, wherein the assigned goal of the agent is*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*expressed as a policy.*

2795. Cisco rACL discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

**Syntax**

A receive ACL is applied with the following global configuration command to distribute the rACL to each LC in the router.

```
[no] ip receive access-list <num>
```

In this syntax, <num> is defined as follows.

```
<1-199> IP access list (standard or extended)
```

```
<1300-2699> IP expanded access list (standard or extended)
```

**GSR: Receive Access Control Lists at 4****3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2796. Cisco rACL discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

Identify protocols used in the network with a classification ACL.

Deploy an rACL that permits all the known protocols that access the GRP. This “discovery” rACL should have both source and destination addresses set to any. Logging can be used to develop a list of source addresses that match the protocol permit statements. In addition to the protocol permit statement, a permit any any log line at the end of the rACL can be used to identify other protocols that would be filtered by the rACL and that might require access to the GRP.

The objective is to determine what protocols the specific network uses. Logging should be used for analysis to determine “what else” might be communicating with the router.

Note: Although the log keyword provides valuable insight into the details of ACL hits, excessive hits to an ACL entry that uses this keyword might result in an overwhelming number of log entries and possibly high router CPU usage. Use the log keyword for short periods of time and only when needed to help classify traffic.

Review identified packets and begin to filter access to the GRP.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Once the packets filtered by the rACL in step 1 have been identified and reviewed, deploy an rACL with a permit any any statement for the allowed protocols. Just as in step 1, the log keyword can provide more information about the packets that match the permit entries. Using deny any any log at the end can help identify any unexpected packets destined to the GRP. This rACL will provide basic protection and will allow network engineers to ensure that all required traffic is permitted.

The objective is to test the range of protocols that need to communicate with the router without having the explicit range of IP source and destination addresses.

GSR: Receive Access Control Lists at 7–8; *see also id.* at 6.

- b. *constructing a topological representation of the computer network from the information.*

2797. Cisco rACL discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

#### Receive Adjacencies and Punted Packets

As described earlier in this document, some packets require GRP processing. The packets are punted from the data-forwarding plane to the GRP. This is a list of the common forms of Layer 3 data that require GRP access.

#### Routing protocols

Multicast control traffic (OSPF, Hot Standby Router Protocol [HSRP], Tag Distribution Protocol [TDP], Protocol Independent Multicast [PIM], and such)

#### Multiprotocol Label Switching (MPLS) packets needing fragmentation

Packets with certain IP options such as router alert

First packet of multicast streams

Fragmented ICMP packets that require reassembly

All traffic destined to the router itself (except for the traffic handled on the LC)

Since rACLs apply to receive adjacencies, the rACL filters some traffic that is not punted to the GRP but is a receive adjacency. The most common example of this is an ICMP echo request (ping). ICMP echo requests directed to the router are handled by the LC CPU; since the requests are receive adjacencies, they are also filtered by the rACL. Therefore, to allow pings to the interfaces (or loopbacks) of the router, the rACL must explicitly permit the echo requests.

Receive adjacencies can be viewed using the show ip cef command.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

GSR: Receive Access Control Lists at 6–7

2798. rACLs may be configured to filter Multicast control traffic such as OSPF protocol messages. NetFuel contends OSPF builds a topological representation of the computer network. Under NetFuel's application of this limitation, Cisco Receive ACL satisfies this limitation.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2799. Cisco rACL discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

Receive Adjacencies and Punted Packets

As described earlier in this document, some packets require GRP processing. The packets are punted from the data-forwarding plane to the GRP. This is a list of the common forms of Layer 3 data that require GRP access.

Routing protocols

Multicast control traffic (OSPF, Hot Standby Router Protocol [HSRP], Tag Distribution Protocol [TDP], Protocol Independent Multicast [PIM], and such)

Multiprotocol Label Switching (MPLS) packets needing fragmentation

Packets with certain IP options such as router alert

First packet of multicast streams

Fragmented ICMP packets that require reassembly

All traffic destined to the router itself (except for the traffic handled on the LC)

Since rACLs apply to receive adjacencies, the rACL filters some traffic that is not punted to the GRP but is a receive adjacency. The most common example of this is an ICMP echo request (ping). ICMP echo requests directed to the router are handled by the LC CPU; since the requests are receive adjacencies, they are also filtered by the rACL. Therefore, to allow pings to the interfaces (or loopbacks) of the router, the rACL must explicitly permit the echo requests.

Receive adjacencies can be viewed using the show ip cef command.

GSR: Receive Access Control Lists at 6–7

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2800. rACLs may be configured to filter Multicast control traffic such as OSPF protocol messages. NetFuel contends OSPF uses a numerical method to model the computer network. Under NetFuel's application of this limitation, Cisco Receive ACL satisfies this limitation.

2801. Additionally this limitation is rendered obvious in view of NetRanger. *See* Ex. A-8, limitation 4.0.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2802. Cisco rACL discloses claim 6, "[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm." '730 Pat. at Cl. 6. For example:

Receive Adjacencies and Punted Packets

As described earlier in this document, some packets require GRP processing. The packets are punted from the data-forwarding plane to the GRP. This is a list of the common forms of Layer 3 data that require GRP access.

Routing protocols

Multicast control traffic (OSPF, Hot Standby Router Protocol [HSRP], Tag Distribution Protocol [TDP], Protocol Independent Multicast [PIM], and such)

Multiprotocol Label Switching (MPLS) packets needing fragmentation

Packets with certain IP options such as router alert

First packet of multicast streams

Fragmented ICMP packets that require reassembly

All traffic destined to the router itself (except for the traffic handled on the LC)

Since rACLs apply to receive adjacencies, the rACL filters some traffic that is not punted to the GRP but is a receive adjacency. The most common example of this is an ICMP echo request (ping). ICMP echo requests directed to the router are handled by the LC CPU; since the requests are receive adjacencies, they are also filtered by the rACL. Therefore, to allow pings to the interfaces (or loopbacks) of the router, the rACL must explicitly permit the echo requests.

Receive adjacencies can be viewed using the show ip cef command.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

GSR: Receive Access Control Lists at 6–7

2803. rACLs may be configured to filter Multicast control traffic such as OSPF protocol messages. NetFuel contends OSPF uses a numerical method to model the computer network. Under NetFuel's application of this limitation, Cisco Receive ACL satisfies this limitation.

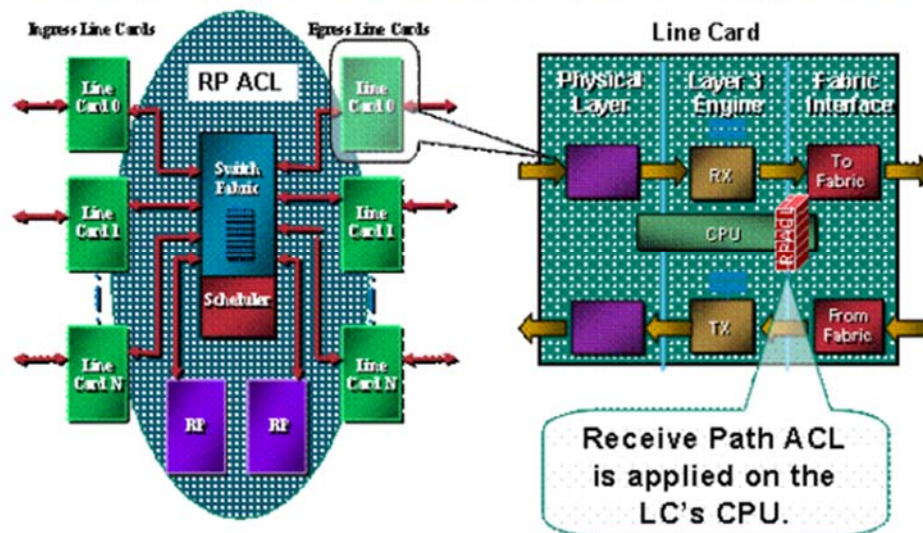
2804. Additionally this limitation is rendered obvious in view of NetRanger. *See* Ex. A-8, limitation 6.0.

6. **Claim 7**

a. *A computer network, comprising:*

2805. Cisco rACL discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

As this image shows, the rACL is executed on each LC before the packet is transmitted to the GRP.



GSR: Receive Access Control Lists at 2

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2806. Cisco rACL discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

2807. Cisco rACL discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

2808. Cisco rACL discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

2809. Cisco rACL discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

2810. Cisco rACL discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2811. Cisco rACL discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

Introduction

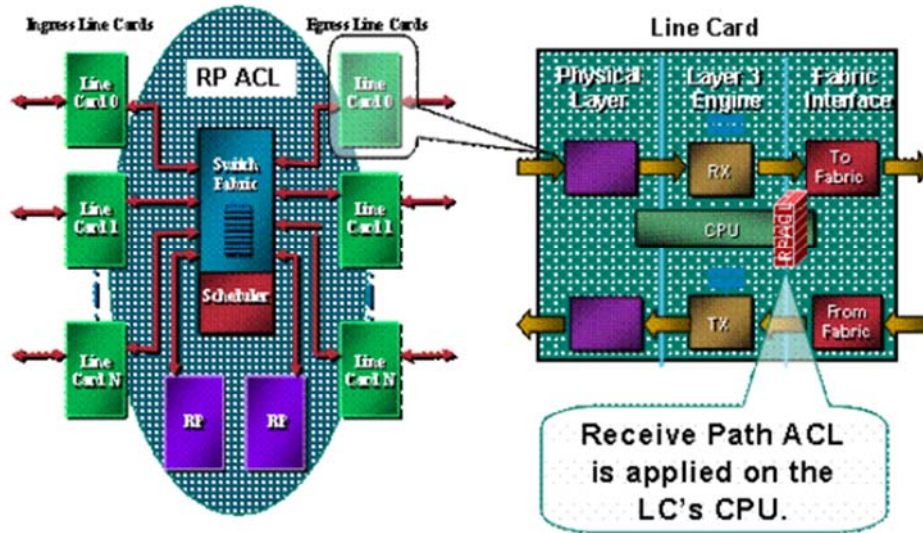
This document describes a new security feature called receive access control lists (rACLs) 1 and presents recommendations and guidelines for rACL deployments. Receive ACLs are used to increase security on Cisco 12000 routers by protecting the router's gigabit route processor (GRP) from unnecessary and potentially nefarious traffic. Receive ACLs were added as a special waiver to the maintenance throttle for Cisco IOS ® Software Release 12.0.21S2 and integrated into Cisco IOS Software Release 12.0(22)S.

GSR: Receive Access Control Lists at 1



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

As this image shows, the rACL is executed on each LC before the packet is transmitted to the GRP.



GSR: Receive Access Control Lists at 2

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2812. Cisco rACL discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2813. Cisco rACL discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2814. Cisco rACL discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2815. Cisco rACL discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2816. Cisco rACL discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2817. Cisco rACL discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**8. Claim 11**

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

2818. Cisco rACL discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

2819. The rACL software may be distributed among different line cards and controlled by a GRP. Under NetFuel’s interpretation of this claim, one rACL software agent may “communicate directly with another” rACL.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2820. To the extent the rACL software fails to explicitly or implicitly disclose a “software agent” that is capable of communicating with other software agents via a communications mechanism, it would have been obvious to in view of Kasteleijn. *See* Ex. A-2 at limitation 11.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2821. Cisco rACL discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

2822. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 12.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2823. Cisco rACL discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

Introduction

This document describes a new security feature called receive access control lists (rACLs) 1 and presents recommendations and guidelines for rACL deployments. Receive ACLs are used to increase security on Cisco 12000 routers by protecting the router's gigabit route processor (GRP) from unnecessary and potentially nefarious traffic. Receive ACLs were added as a special waiver to the maintenance throttle for Cisco IOS ® Software Release 12.0.21S2 and integrated into Cisco IOS Software Release 12.0(22)S.

GSR: Receive Access Control Lists at 1

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

2824. Cisco rACL discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

2825. IOS controls the operation of rACL agents based at least on CLI commands and configuration files, which are predetermined criteria under NetFuel’s application of the claims.

2826. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 17.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2827. Cisco rACL discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

Cisco IOS instantiates the rACL software and thus “spawns” it under NetFuel’s application of the claims.

Traffic that enters an LC is first sent to the local CPU of the LC, and packets that require processing by the GRP are queued for forwarding to the route processor. The receive ACL is created on the GRP and then pushed down to the CPUs of the various LCs. Before traffic is sent from the LC CPU to the GRP, the traffic is compared to the rACL. If permitted, the traffic passes to the GRP, while all other traffic is denied. The rACL is inspected prior to the LC to GRP rate-limiting function. Since the rACL is used for all receive adjacencies, some packets that are handled by the LC CPU (such as echo requests) are subject to rACL filtering as well. This needs to be taken into account when designing rACL entries.

GSR: Receive Access Control Lists at 3

2828. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 18.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*environment.*

2829. Cisco rACL discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

2830. NetFuel contends this limitation is inherent in IOS. Under NetFuel’s application of the claims, this limitation is anticipated or rendered obvious by Cisco IOS with rACL capability.

2831. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 19.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2832. Cisco rACL discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

**Performance Impact**

No memory is consumed other than that necessary to hold the single configuration entry and the defined access list itself. The rACL is copied to each LC, so a slight area of memory is taken on each LC. Overall, resources utilized are minuscule, especially when compared with the benefits of deployment.

GSR: Receive Access Control Lists at 3.

2833. Additionally, NetFuel contends this limitation is inherent in IOS. Under NetFuel’s application of the claims, this limitation is anticipated or rendered obvious by Cisco IOS with rACL capability.

2834. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 19.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism includes the graphical user interface.*

2835. Cisco rACL discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

2836. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 22.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2837. Cisco rACL discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2838. Cisco rACL discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

2839. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 26.0.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2840. Cisco rACL discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

2841. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 29.0.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

19. *Claim 30*

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2842. Cisco rACL discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

Introduction

This document describes a new security feature called receive access control lists (rACLs) 1 and presents recommendations and guidelines for rACL deployments. Receive ACLs are used to increase security on Cisco 12000 routers by protecting the router's gigabit route processor (GRP) from unnecessary and potentially nefarious traffic. Receive ACLs were added as a special waiver to the maintenance throttle for Cisco IOS ® Software Release 12.0.21S2 and integrated into Cisco IOS Software Release 12.0(22)S.

GSR: Receive Access Control Lists at 1

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2843. Cisco rACL discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2844. Cisco rACL discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2845. Cisco rACL discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2846. Cisco rACL discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2847. Cisco rACL discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

2848. Cisco rACL discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2849. Cisco rACL discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

2850. Cisco rACL discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2851. Cisco rACL discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2852. Cisco rACL discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

2853. Cisco rACL discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2854. Cisco rACL discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2855. Cisco rACL discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2856. Cisco rACL discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2857. Cisco rACL discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2858. Cisco rACL discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2859. Cisco rACL discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

**V. Anticipation by and/or Obviousness in View of the IBM xSeries Server Software Rejuvenation Agent, marketed by IBM by 2000 at the latest.**

2860. As explained in detail below and in the chart attached as Ex. A-12, IBM xSeries Server Software Rejuvenation Agent anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2861. IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

“The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

Castilli at 313.

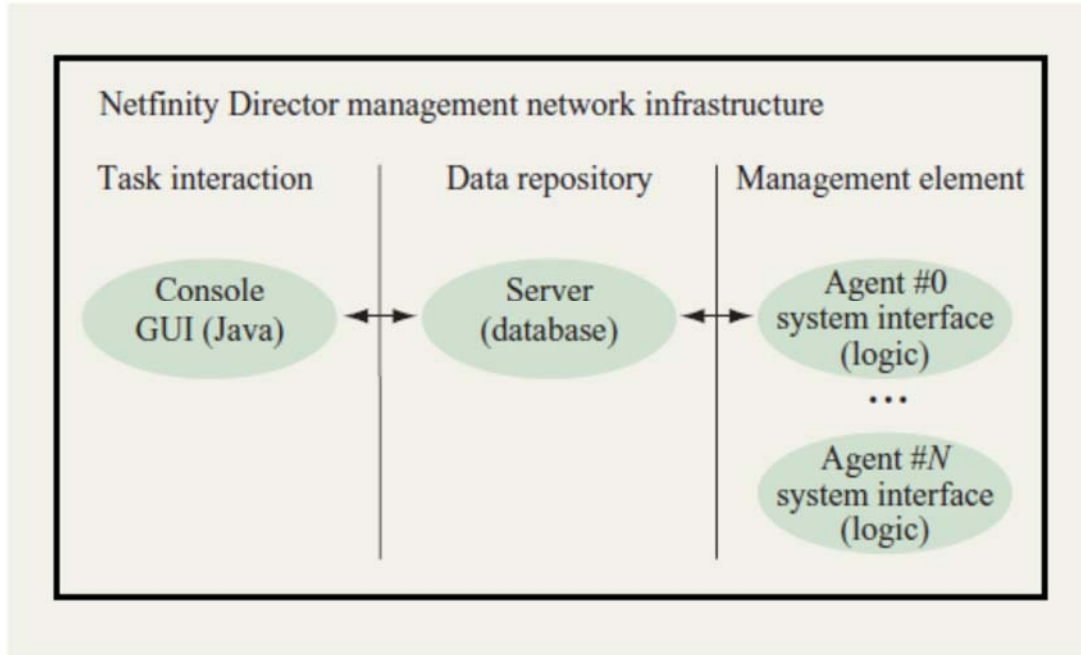


Figure 1

IBM Director framework.

Castelli at 315; *see also id.* at 311, 12, 22-23

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2862. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

### 3. The xSeries Software Rejuvenation Agent

The xSeries Software Rejuvenation Agent (SRA) was designed to monitor consumable resources, estimate the time to exhaustion of those resources, and generate alerts to the management infrastructure when the time to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

exhaustion is less than a user-defined notification horizon. The management infrastructure provides a graphical user interface for the user to configure the SRA, and accepts and acts upon the alerts as described below.

The SRA was designed according to a number of ground rules, with the prime objective of maximizing flexibility, portability, and customer acceptance. For maximum generality and user acceptance, no modification to the application is allowed, to support either failure prediction or rejuvenation. Similarly, no access to the kernel is allowed, in order to facilitate error containment, cross-OS portability, and customer acceptance. The agent must use published and architected interfaces for data acquisition, alerting, and rejuvenation in order to minimize sensitivity to gratuitous interface churn and provide a product that is relatively stable across multiple generations of operating systems and applications. ***The agent must be relatively portable across operating systems to allow us to economically attack the different markets of commercial interest to IBM.*** A simple user interface is required which contains a minimum number of tunable parameters and is easy to set up and understand. Finally, because in many cases we do not know in advance which resources will be exhausted in the myriad environments in which we will be using SRA, the agent must be able to adapt to monitor new exhaustible parameters and execute new algorithms to predict exhaustion of those resources. These considerations caused us to partition the design into an OS-dependent data acquisition subsystem, a portable analytical subsystem with highly configurable input parameters, an architected management interface, and a management infrastructure that spans multiple IBM-supported operating systems. The SRA is incorporated as a component of IBM Director, which is discussed next.

Castelli at 314-315; *see also id.* 16, 17, 18.

- c. *is able to communicate with other software agents in the computer network;*

2863. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim

1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl.

1. For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. All notifications are done with the IBM Director event driven notification mechanism. These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

Castelli at 317.

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

interface to a managed object, which in our case is a server or cluster of xSeries servers. Events provide a notification mechanism from an agent into the IBM Director environment.

Castelli at 315.

2864. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. *See* Ex. A-2 at limitation 1.2.

d. *is capable of perceiving its own state*

2865. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

---

Castelli at 315.

The agents *reside on each managed system (such as an xSeries server)* and act as passive, nonintrusive *native applications*. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

Castelli at 315.

***Management console:***

The IBM Director management console is the graphical user interface (GUI) from which administrative tasks are performed. It is the primary interface with the administrator, and is used to configure the SRA as described below. The management console GUI is Java-based, with all state information stored on the server. ***It runs as a locally installed Java application in a Java Virtual Machine (JVM\*\*).***

Castelli at 315.

e. *and is able to clone itself,*

2866. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

Software rejuvenation is presented to the user as a highly stylized means to schedule rejuvenation. The user has at his disposal the building blocks of a schedule and a set of resources. The resources may be scheduled for time-based rejuvenation or configured for exhaustion forecasting.

The SRA user interface is presented to the user in the form of a calendar (Figure 4) in which the user conceptually can see and manipulate rejuvenations. To schedule a rejuvenation for a certain day, the user drags and drops a node of the cluster from the left-hand side of the interface (e.g., “platini” and “zico” are servers within cluster “copamundial”) onto the day

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

of the week when rejuvenation is desired. A follow-up dialogue (Figure 5) then negotiates whether the user wishes the rejuvenation to occur daily, weekly, monthly, or on some other periodic basis, and the time at which the rejuvenation is to occur. The user can designate certain days of the week as invalid, when no rejuvenation can occur, and multiple rejuvenations are prohibited from being scheduled at the same time. Among other rejuvenation options, the user can engage cluster-specific logic designed to ensure that a properly planned failover can occur. As shown in the lefthand side of Figure 6, the user can ask the rejuvenation logic to confirm that at least one backup node exists which can handle the failover workload prior to rejuvenating ("check for one"), ensure that all backup nodes can handle the workload ("check for all"), or rejuvenate without checking ("skip check"). If one of the first two options is selected and a backup node cannot be found, rejuvenation is postponed until the next opportunity. The proactive option of exhaustion prediction is provided to evoke an advanced and noninteractive scheme for scheduling resources for rejuvenation. The user can set up a stand-alone system or a clustered system for this level of support. In the expert mode of operation for exhaustion prediction, the user can allow an application to schedule rejuvenation automatically without his interaction. The user configures a cluster or a node for prediction capabilities by invoking a simple prediction configuration menu (Figure 7), and selecting the notification horizon and the type of action desired when exhaustion is predicted to occur within that horizon. Very few other parameters are configurable by the user.

Castelli at 316-317

To the extent this is not explicitly or implicitly disclosed in Castelli, it would have been obvious for a person of ordinary skill in the art at the time of invention to modify Castelli so that the SRA policy established for one node or cluster may be copied to another node/cluster, and therefore the agent responsible for that node/cluster. The motivation of this modification would be to reduce setup time and redundancy.

2867. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. See Ex. A-2 at limitation 1.2.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2868. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, "and wherein the goal is a programmatic expression of a predefined task for the software agent."

'730 Pat. at Cl. 1. For example:

Software rejuvenation is presented to the user as a highly stylized means to schedule rejuvenation. The user has at his disposal the building blocks of a schedule and a set of resources. The resources may be scheduled for time-based rejuvenation or configured for exhaustion forecasting.

The SRA user interface is presented to the user in the form of a calendar (Figure 4) in which the user conceptually can see and manipulate

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

rejuvenations. To schedule a rejuvenation for a certain day, the user drags and drops a node of the cluster from the left-hand side of the interface (e.g., “platini” and “zico” are servers within cluster “copamundial”) onto the day of the week when rejuvenation is desired. A follow-up dialogue (Figure 5) then negotiates whether the user wishes the rejuvenation to occur daily, weekly, monthly, or on some other periodic basis, and the time at which the rejuvenation is to occur. The user can designate certain days of the week as invalid, when no rejuvenation can occur, and multiple rejuvenations are prohibited from being scheduled at the same time. Among other rejuvenation options, the user can engage cluster-specific logic designed to ensure that a properly planned failover can occur. As shown in the lefthand side of Figure 6, the user can ask the rejuvenation logic to confirm that at least one backup node exists which can handle the failover workload prior to rejuvenating (“check for one”), ensure that all backup nodes can handle the workload (“check for all”), or rejuvenate without checking (“skip check”). If one of the first two options is selected and a backup node cannot be found, rejuvenation is postponed until the next opportunity. The proactive option of exhaustion prediction is provided to evoke an advanced and noninteractive scheme for scheduling resources for rejuvenation. The user can set up a stand-alone system or a clustered system for this level of support. In the expert mode of operation for exhaustion prediction, the user can allow an application to schedule rejuvenation automatically without his interaction. The user configures a cluster or a node for prediction capabilities by invoking a simple prediction configuration menu (Figure 7), and selecting the notification horizon and the type of action desired when exhaustion is predicted to occur within that horizon. Very few other parameters are configurable by the user.

Castelli at 316-317.

g. *monitoring the computer network;*

2869. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim

1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

*“IBM Director agents:*

The agents ***reside on each managed system (such as an xSeries server)*** and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

***In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes.*** These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:

*Inventory:* IBM Director discovers new managed systems, collects the appropriate information about these systems, and stores it in the inventory database. It can then be viewed through either a default or a customized view.

*Resource monitors:* Resource monitors (**Figure 2**) enable the user to view statistics and usage of critical resources on the network. Information can be

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

collected and monitored on attributes such as CPU, disk, memory, and network. SRA is a specialized instance of a resource monitor. *Event management:* Event management (**Figure 3**) enables the user to view a log of events that have occurred for a managed system or group of systems and to create event action plans to associate an event with a desired action, such as sending an e-mail, starting a program, logging to a file, or invoking rejuvenation. When the SRA has detected an impending resource exhaustion, it generates events that can be viewed using this functionality.

Castelli at 315-316, *see also id.* 312, 14, 17

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2870. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

Prediction algorithms

In the current version of SRA, rejuvenation can be based on elapsed time since the last rejuvenation, or on prediction of impending exhaustion.

When using timed rejuvenation, the user interface of IBM Director is used to schedule and perform rejuvenation at a period specified by the user. A calendar interface allows the user to select when to rejuvenate different nodes of the cluster, and to select “blackout” times during which no rejuvenation is to be allowed. Although this sounds rather primitive, our analysis (presented below) shows that for typical clusters that undergo aging, system availability can be improved significantly via this technique.

Castelli at 317.

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

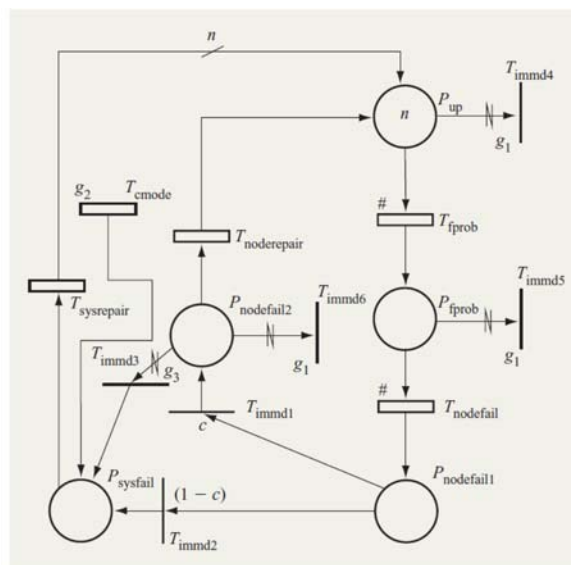
Castelli at 317-318, *see also id.* at 323, 24, 27, 28, 29.

Additionally, Castelli discloses this limitation under NetFuel's interpretation of this term. The initial parameters of the SRA agents are test policies. Execution and enforcement of these test policies then models and reflects the actual behavior of the computer network, which can then be monitored by a user or other software that can then instantiate a revised optimal policy.

- i. *including predicting a failure of a network component based on a prediction algorithm*

2871. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 22**

Basic cluster system.

Castelli at 327

“**Figure 22** shows the basic model of our cluster system. The cluster consists of  $n$  nodes. Initially, all of the nodes are in a “robust” working state, indicated by tokens in place  $P_{up}$ , in which the probability of node failure is zero. As time progresses, each node eventually transits to a “failure-probable” state (place  $P_{fprob}$ ) through the transition  $T_{fprob}$ . The nodes are still operational in this state but can fail (transit to place  $P_{nodefail1}$  with a nonzero probability). If a node crashes, it can recover with a probability  $c$  through the transition  $T_{noderepair}$ . In this case, the node goes back to the place  $P_{up}$  which represents the clean state of the node. The node recovery can fail with a probability  $(1 - c)$ , leading to a system failure (all  $n$  nodes are down). Place  $P_{sysfail}$  represents this system-level failure state.”

Castelli at 327



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

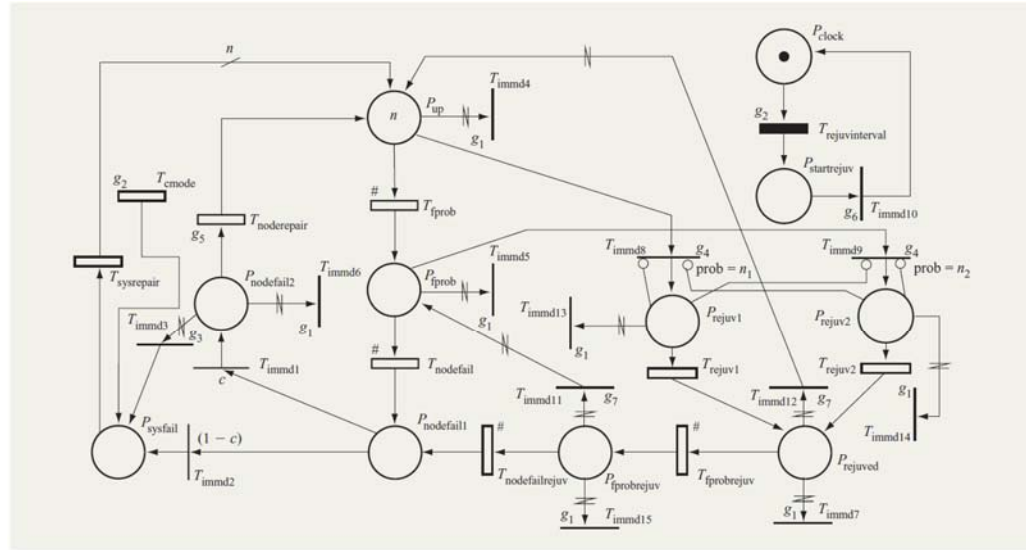


Figure 23

Cluster system employing simple time-based rejuvenation.

Castelli at 328, *see also id.* at 329.

j. wherein said modeling comprises determining appropriate policy based on the prediction; and

2872. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim

1, “wherein said modeling comprises determining appropriate policy based on the prediction.”

'730 Pat. at Cl. 1. For example:

***Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.*** The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the ***agent automatically performs the analysis.***

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and *the analysis and extrapolation over the three-day horizon are performed using that one day's worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function* without overfitting the data.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The *prediction algorithm fits several types of curves to the data in the fitting window*; these curves have been selected for their ability to capture different types of temporal trends. *A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon.* Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, *see also id.* 323, 24, 27, 28, 29.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

2873. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example:

*Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.* The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the *agent automatically performs the analysis.*

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and *the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function* without overfitting the data.

The *prediction algorithm fits several types of curves to the data in the fitting window*; these curves have been selected for their ability to capture different types of temporal trends. *A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon.* Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, *see also id.* 323, 24, 27, 28, 29

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2874. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim

1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

***Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data.*** The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the ***agent automatically performs the analysis.***

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and ***the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function*** without overfitting the data.

The ***prediction algorithm fits several types of curves to the data in the fitting window***; these curves have been selected for their ability to capture different types of temporal trends. ***A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon.*** Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318.

More advanced approaches are now available, such as the IBM Secureway Network Dispatcher [32]. This product provides enhanced IP-level load-balancing mechanisms and content-based routing, as well as improved management and availability functions. **Figure 17** illustrates the network dispatcher operations for a LAN implementation. ***As part of load***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*balancing, the dispatcher's scheduling policy is dynamically based on each server's load and availability. This is partly accomplished by having each server send periodic utilization information to the dispatcher.* This utilization information can easily be augmented by health information in the form of time until resource exhaustion, degree of resource exhaustion or, in its simplest form, time remaining until a timed rejuvenation. This health information can be sent to the dispatcher in order to schedule actions for individual servers. The scheduling of these actions can also take into account aggregate loading of the web host.

Castelli at 323; *see also id.* 324, 27, 28, 29.

2875. To the extent this limitation is not explicitly or implicitly disclosed by Castelli, it would have been obvious in view of Turek. *See* Ex. A-1 at limitation 1.11.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2876. IBM xSeries Server Software Rejuvenation Agent discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

3. The xSeries Software Rejuvenation Agent

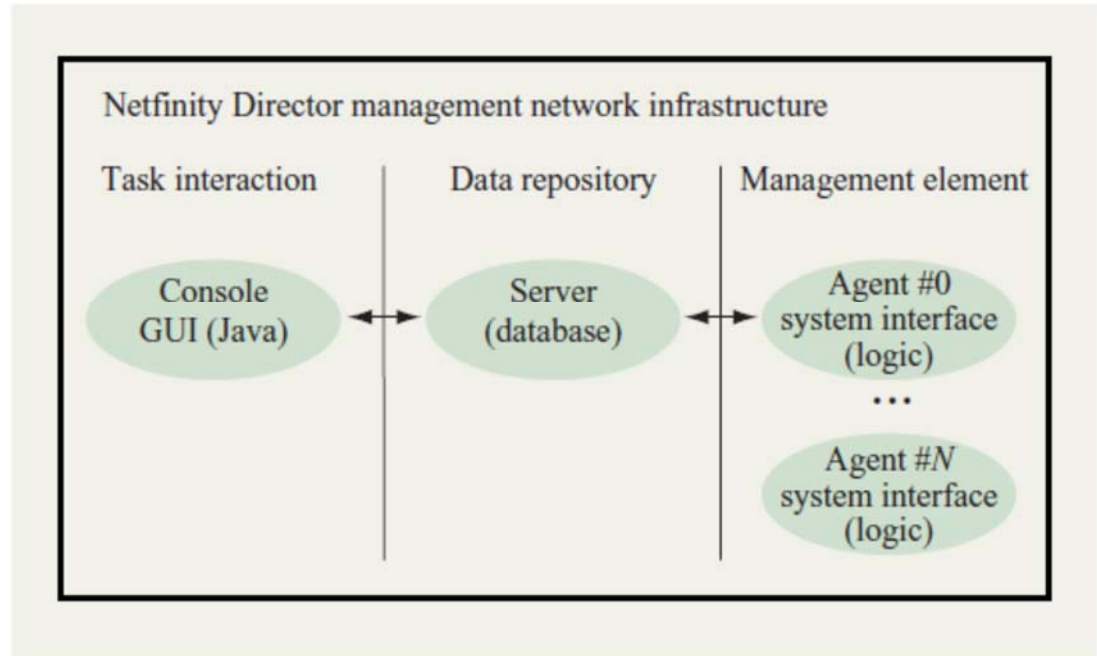
The xSeries Software Rejuvenation Agent (SRA) was designed to monitor consumable resources, estimate the time to exhaustion of those resources, and generate alerts to the management infrastructure when the time to exhaustion is less than a user-defined notification horizon. The management infrastructure provides a graphical user interface for the user to configure the SRA, and accepts and acts upon the alerts as described below.

The SRA was designed according to a number of ground rules, with the prime objective of maximizing flexibility, portability, and customer acceptance. For maximum generality and user acceptance, no modification to the application is allowed, to support either failure prediction or rejuvenation. Similarly, no access to the kernel is allowed, in order to facilitate error containment, cross-OS portability, and customer acceptance. The agent must use published and architected interfaces for data acquisition, alerting, and rejuvenation in order to minimize sensitivity to gratuitous interface churn and provide a product that is relatively stable across multiple generations of operating systems and applications. ***The agent must be relatively portable across operating systems to allow us to economically attack the different markets of commercial interest to IBM.*** A simple user interface is required which contains a minimum number of tunable parameters and is easy to set up and understand. Finally, because in many cases we do not know in advance which resources will be exhausted in the myriad environments in which we will be using SRA, the agent must be able to adapt to monitor new exhaustible parameters and execute new algorithms to predict exhaustion of those resources. These considerations

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

caused us to partition the design into an OS-dependent data acquisition subsystem, a portable analytical subsystem with highly configurable input parameters, an architected management interface, and a management infrastructure that spans multiple IBMsupported operating systems. The SRA is incorporated as a component of IBM Director, which is discussed next.

Castelli at 314-315.



**Figure 1**

IBM Director framework.

Castelli at 315; *see also id.* 316, 317, 318, 323, 24, 27, 28, 29

**3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2877. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

*IBM Director agents:*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The agents *reside on each managed system (such as an xSeries server)* and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

*In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes.* These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:

*Inventory:* IBM Director discovers new managed systems, collects the appropriate information about these systems, and stores it in the inventory database. It can then be viewed through either a default or a customized view.

*Resource monitors:* Resource monitors (**Figure 2**) enable the user to view statistics and usage of critical resources on the network. Information can be collected and monitored on attributes such as CPU, disk, memory, and network. SRA is a specialized instance of a resource monitor.

*Event management:* Event management (**Figure 3**) enables the user to view a log of events that have occurred for a managed system or group of systems and to create event action plans to associate an event with a desired action, such as sending an e-mail, starting a program, logging to a file, or invoking rejuvenation. When the SRA has detected an impending resource exhaustion, it generates events that can be viewed using this functionality.

pp. 315-316; *see id.* at 312, 14, 17.

- b. *constructing a topological representation of the computer network from the information.*

2878. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

As discussed above, it would have been obvious to combine Castelli with the OSFP routing protocol (RFC 2328). Under NetFuel’s interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. Thus it would have been obvious for SRA agents to also monitor network resources used by OSFP, and to use OSPF routing protocol. OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement Castelli’s system on network devices using OSPF.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*method.*

2879. IBM xSeries Server Software Rejuvenation Agent discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day’s worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, *see also id.* at 323, 327, 328, 329.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2880. IBM xSeries Server Software Rejuvenation Agent discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

Single-parameter predictive rejuvenation relies on curve-fitting analysis and projection, using recently observed data. The projected data is compared to prespecified upper and lower exhaustion thresholds within a notification time horizon. The user specifies the notification horizon and the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

desired parameters (some parameters believed to be highly indicative are always monitored by default), and the agent automatically performs the analysis.

The curve-fitting algorithm operates on a sliding window of data spanning a temporal interval which is a fixed fraction (say, 1/3) of the notification horizon. For example, if the user wishes to be informed or have rejuvenation invoked if exhaustion is projected to occur within, say, three days, the data window is set to one day, and the analysis and extrapolation over the three-day horizon are performed using that one day's worth of data. The sampling interval is selected to provide enough data points within the fitting window to allow the prediction algorithm to adequately smooth the data and select an appropriate prediction function without overfitting the data.

The prediction algorithm fits several types of curves to the data in the fitting window; these curves have been selected for their ability to capture different types of temporal trends. A model-selection criterion is applied to choose the best prediction curve, which is then extrapolated to the user-specified horizon. Several parameters that are indicative of resource exhaustion are monitored and extrapolated independently. If any monitored parameter exceeds the specified minimum or maximum value within the horizon, a request to rejuvenate is sent to the management infrastructure. In most cases, it is also possible to identify the process that is consuming the preponderance of the resource being exhausted, in order to support selective rejuvenation, as described below. Details of the curve fitting and model selection are given in Appendix A.

Castelli at 317-318, *see also id.* at 323, 324, 327, 328, 329.

6. ***Claim 7***

a. *A computer network, comprising:*

2881. IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim

7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

The main contribution of this paper is the development of a methodology for proactive management of software systems which are prone to aging, and specifically to resource exhaustion. The application of software rejuvenation for cluster systems is by itself a novel contribution.

Castelli at 313.

---

p. 315.

Software failure prediction and rejuvenation in a cluster environment

A cluster is a collection of independent, self-contained computer systems working together to provide a more reliable and powerful system than a single node by itself

p. 312.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

This technology has been incorporated into the IBM Director for xSeries servers.

p. 311.

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. ***This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000.*** Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

p. 323.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2882. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

2883. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

2884. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

2885. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

f. *is able to clone itself;*

2886. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

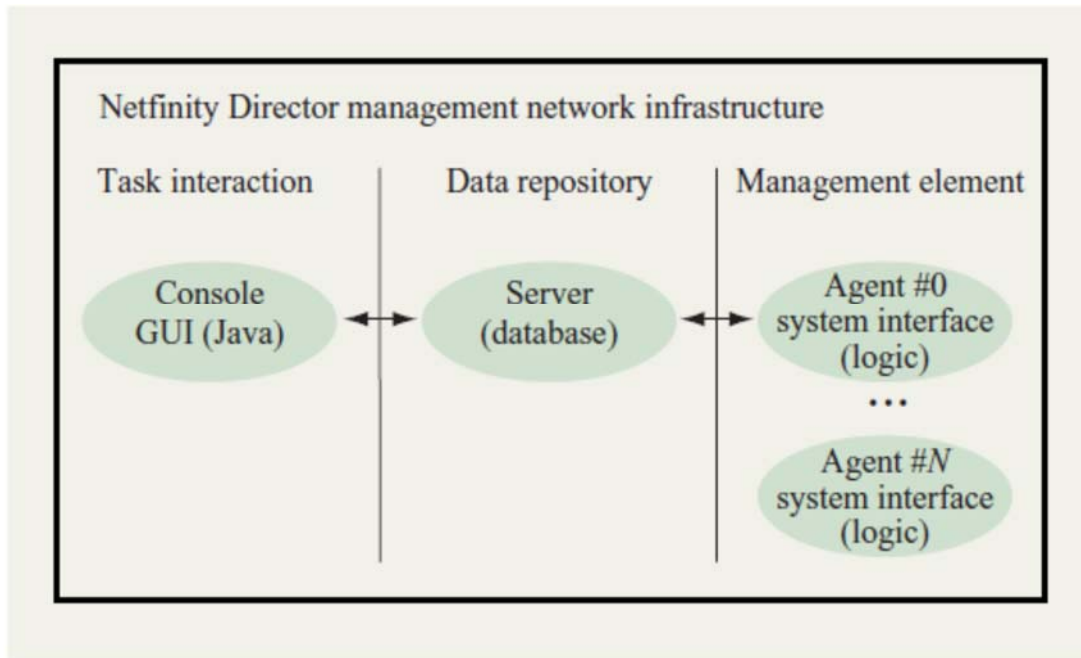
2887. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

3. The xSeries Software Rejuvenation Agent

The xSeries Software Rejuvenation Agent (SRA) was designed to monitor consumable resources, estimate the time to exhaustion of those resources, and generate alerts to the management infrastructure when the time to exhaustion is less than a user-defined notification horizon. The management infrastructure provides a graphical user interface for the user to configure the SRA, and accepts and acts upon the alerts as described below.

The SRA was designed according to a number of ground rules, with the prime objective of maximizing flexibility, portability, and customer acceptance. For maximum generality and user acceptance, no modification to the application is allowed, to support either failure prediction or rejuvenation. Similarly, no access to the kernel is allowed, in order to facilitate error containment, cross-OS portability, and customer acceptance. The agent must use published and architected interfaces for data acquisition, alerting, and rejuvenation in order to minimize sensitivity to gratuitous interface churn and provide a product that is relatively stable across multiple generations of operating systems and applications. ***The agent must be relatively portable across operating systems to allow us to economically attack the different markets of commercial interest to IBM.*** A simple user interface is required which contains a minimum number of tunable parameters and is easy to set up and understand. Finally, because in many cases we do not know in advance which resources will be exhausted in the myriad environments in which we will be using SRA, the agent must be able to adapt to monitor new exhaustible parameters and execute new algorithms to predict exhaustion of those resources. These considerations caused us to partition the design into an OS-dependent data acquisition subsystem, a portable analytical subsystem with highly configurable input parameters, an architected management interface, and a management infrastructure that spans multiple IBM-supported operating systems. The SRA is incorporated as a component of IBM Director, which is discussed next.

pp. 314-315.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 1**

IBM Director framework.

p. 315; *see also id.* 316, 317, 318, 323, 24, 27, 28, 29

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2888. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2889. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein the modeler determines appropriate policy based on the prediction;*

2890. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2891. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2892. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2893. IBM xSeries Server Software Rejuvenation Agent discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

2894. IBM xSeries Server Software Rejuvenation Agent discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2895. IBM xSeries Server Software Rejuvenation Agent discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

A notification mechanism, via the built-in IBM Director’s event mechanism, allows for notification of any impending exhaustion through either a pop-up or ticker tape. ***All notifications are done with the IBM Director event driven notification mechanism.*** These events are used to drive the scheduling of an automatic rejuvenation, of notifications, and of other user-defined actions (such as running a remote program). The status is reported using the IBM Director-provided event log mechanism.

p. 317.

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers. ***Events provide a notification mechanism from an agent into the IBM Director environment.***

p. 315.

To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn.

See Ex. A-2 at limitation 12.0.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*for a particular network component.*

2896. IBM xSeries Server Software Rejuvenation Agent discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

The three tiers of IBM Director are the console, server, and agent. The console provides a Java\*\*-based interface for accessing the functionality (via a set of tasks) of the IBM Director environment. ***The server controls access to the function, data, and agents for a given task, and the agent is the interface to a managed object, which in our case is a server or cluster of xSeries servers.*** Events provide a notification mechanism from an agent into the IBM Director environment.

p. 315.

***IBM Director agents:***

The agents ***reside on each managed system (such as an xSeries server)*** and act as passive, nonintrusive ***native applications***. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

p. 315.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2897. IBM xSeries Server Software Rejuvenation Agent discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn. *See* Ex. A-2 at limitation 17.0.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2898. IBM xSeries Server Software Rejuvenation Agent discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn.

See Ex. A-2 at limitation 18.0.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2899. IBM xSeries Server Software Rejuvenation Agent discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Kasteleijn.

See Ex. A-2 at limitation 18.0.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2900. IBM xSeries Server Software Rejuvenation Agent discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

*Management server:* The management server is the platform used for the central management server, where management databases, the server engine, and management application logic reside... In addition to the SRA function, IBM Director supports a comprehensive set of tasks for agent nodes. These nodes communicate directly with the IBM Director server, allowing numerous tasks to be performed, of which the following short list is representative:

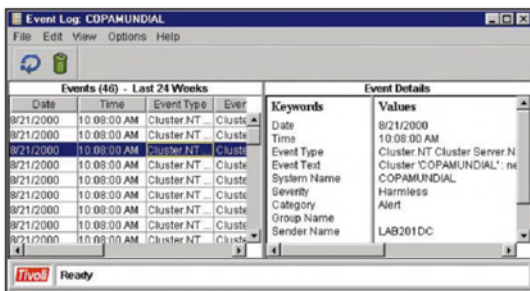
p. 315.

15. ***Claim 22***

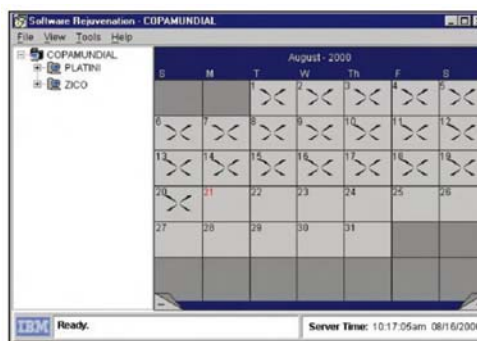
- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2901. IBM xSeries Server Software Rejuvenation Agent discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

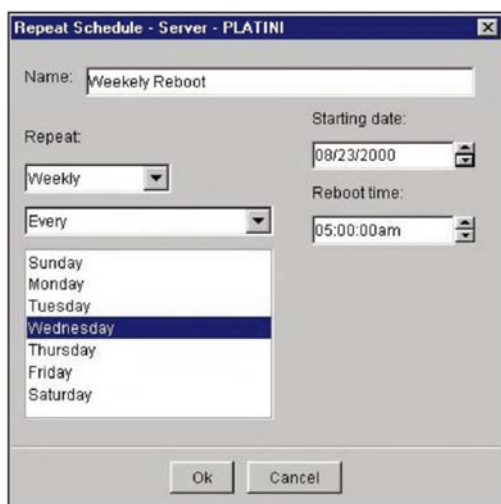


**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Figure 3**

Event management.

**Figure 4**

Software rejuvenation calendar.

**Figure 5**

Scheduling rejuvenation.

Figs. 3-5, p. 316

Software rejuvenation is presented to the user as a highly stylized means to schedule rejuvenation. The user has at his disposal the building blocks of a schedule and a set of resources. The resources may be scheduled for time-based rejuvenation or configured for exhaustion forecasting.

The SRA user interface is presented to the user in the form of a calendar (Figure 4) in which the user conceptually can see and manipulate rejuvenations. To schedule a rejuvenation for a certain day, the user drags and drops a node of the cluster from the left-hand side of the interface (e.g., “platini” and “zico” are servers within cluster “copamundial”) onto the day of the week when rejuvenation is desired. A follow-up dialogue (Figure 5) then negotiates whether the user wishes the rejuvenation to occur daily, weekly, monthly, or on some other periodic basis, and the time at which the rejuvenation is to occur. The user can designate certain days of the week as invalid, when no rejuvenation can occur, and multiple rejuvenations are prohibited from being scheduled at the same time. Among other



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

rejuvenation options, the user can engage cluster-specific logic designed to ensure that a properly planned failover can occur. As shown in the lefthand side of Figure 6, the user can ask the rejuvenation logic to confirm that at least one backup node exists which can handle the failover workload prior to rejuvenating (“check for one”), ensure that all backup nodes can handle the workload (“check for all”), or rejuvenate without checking (“skip check”). If one of the first two options is selected and a backup node cannot be found, rejuvenation is postponed until the next opportunity.

The proactive option of exhaustion prediction is provided to evoke an advanced and noninteractive scheme for scheduling resources for rejuvenation. The user can set up a stand-alone system or a clustered system for this level of support. In the expert mode of operation for exhaustion prediction, the user can allow an application to schedule rejuvenation automatically without his interaction. The user configures a cluster or a node for prediction capabilities by invoking a simple prediction configuration menu (Figure 7), and selecting the notification horizon and the type of action desired when exhaustion is predicted to occur within that horizon. Very few other parameters are configurable by the user.

pp. 316-317.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2902. IBM xSeries Server Software Rejuvenation Agent discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2903. IBM xSeries Server Software Rejuvenation Agent discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious in view of Goldman.

See Ex. A-3 at limitation 26.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

2904. IBM xSeries Server Software Rejuvenation Agent discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

IBM xSeries Server Software Rejuvenation Agent discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.”

’730 Pat. at Cl. 30.

For example:

Management console: The IBM Director management console is the graphical user interface (GUI) from which administrative tasks are performed. It is the primary interface with the administrator, and is used to configure the SRA as described below. The management console GUI is Java-based, with all state information stored on the server. It runs as a locally installed Java application in a Java Virtual Machine (JVM\*\*). Management server: The management server is the platform used for the central management server, where management databases, the server engine, and management application logic reside.

IBM Director agents: The agents reside on each managed system (such as an xSeries server) and act as passive, nonintrusive native applications. The SRA task that collects data, predicts resource exhaustion, and generates events runs as an agent.

p. 315.

This technology has been incorporated into the IBM Director for xSeries servers.

p. 311

We have developed, analyzed, and implemented a framework for detection, prediction, and proactive management of software aging. This technology is applicable to a wide range of operating environments, and has been implemented in the xSeries Software Rejuvenation Agent. It has been commercially available on xSeries servers since the end of 2000. Our cost and availability models indicate that rejuvenation significantly improves cluster system availability and reduces downtime cost.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

p. 323.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2905. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2906. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2907. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2908. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

2909. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l.

- g. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

2910. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

2911. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

2912. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2913. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

2914. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*assigned goal of the agent is expressed as a policy.*

2915. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

2916. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

2917. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

2918. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

2919. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

2920. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2921. IBM xSeries Server Software Rejuvenation Agent discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

**W. Anticipation by and/or Obviousness in View of United States Patent No. 6,826,150, filed 10/2/2001 and issued to Bhattacharya, et al. on 11/30/2004.**

2922. As explained in detail below and in the chart attached as Ex. A-22, Bhattacharya anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

2923. Bhattacharya discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” '730 Pat. at Cl. 1. For example:

*A method* for policing traffic on a ***computer communications network*** having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for ***monitoring*** and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2924. Further examples include Bhattacharya at 1:15–18, 2:5–22, 3:20–24; Bhattacharya at Figs. 1 and 2.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

2925. Bhattacharya discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An ***individual policer is established at each node for monitoring and/or policing the traffic incoming to that node.*** Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and ***when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node.*** The other individual policers or a master policer will receive the exported information. ***-The individual policers police the traffic incoming to its associated node*** depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2926. Further examples include Bhattacharya at 2:5–22, 3:25–45, 3:45–61.

- c. *is able to communicate with other software agents in the computer network;*

2927. Bhattacharya discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

In a preferred embodiment, the policing decision for each packet is made using a modified version of the token bucket algorithm, and including thresholds wherein the information messages between individual policers occurs only when these thresholds are exceeded. Such thresholds may be on the data received and/or time from last up date or combinations of both. ***Moreover, it is possible to interchange the information between the individual policers via channels or a medium outside that of the network***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*itself*. For example, dial up telephone/fax lines, and the like. However, the same principle may be employed to other centralized policing algorithms.

Bhattacharya at 4:20-32.

2928. Further examples include Bhattacharya at Abstract, 8:5-15, 1:15-18, 1:20-60, 3:20-24, 3:45-61, 5:23-28, 3:62-4:10; Bhattacharya at Figs. 1, 2.

d. *is capable of perceiving its own state*

2929. Bhattacharya discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

The inventive system and method implements the policing policy across the entire network, or part thereof, by providing many modules referenced as individual policers. Each individual policer can observe and police only a part of the traffic. In this system *each individual policer uses “global state variables” that stores the measure traffic for the entire traffic-class and “local state variables” that store the measure of the part of traffic that is permitted by this individual policer. Each individual policer exports the measure of traffic it permitted to all individual policers, in the form of the local state variables, or functions thereof.* After exporting such information, the individual policer clears the local state variables. Upon receiving such exported information, all the individual policers update their global state variables. At any time, the global state variables of an individual policer account for the total of all the measure of traffic exported from all individual policers till that time. At any time, the local state variables of an individual policer account for the traffic that was permitted at the same individual policer since it last exported the measure of traffic it permitted.

Bhattacharya at 3:25-45.

2930. Further examples include Bhattacharya at Abstract, 2:5-22, 3:45-61, 3:62-4:10.

e. *and is able to clone itself,*

2931. Bhattacharya discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

When a forwarding engine processes a packet from a directly attached port, a logic module in the individual policer, referenced as policer-logic, uses the packet length, the policer-memory record and the value of the free-running counter, referenced as current time, to perform the computation represented in the following pseudo-code:-



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**


---

```

50  if ((packet_length + byte_count_local + bucket
      - contract_rate*(current_time - last_update_time)) > burst)
      {
        police_packet( );    /* packet out of profile => police */
      }
    else
55  {
        permit_packet( );    /* Packet in profile => permit */
        byte_count_local += packet_length;
      }
    /* decision to send a PUP */
    if (pup_transmit_condition) /* send PUP & update bucket */
    {
60    /* Send PUP to other individual policers */
        transmit_pup(traffic_class, byte_count_local);
        /* update bucket in this individual policer */
        bucket += (byte_count_local
                  - contract_rate*(current_time
                                - last_update_time));
65    if (bucket < 0)
        bucket = 0;

```

---

-continued

---

```

last_update_time = current_time;
byte_count_local = 0;
}

```

---

5

Bhattacharya at 6:41-7:5.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

2932. Bhattacharya discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. ***Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer.*** Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2933. Further examples include Bhattacharya at 2:5–22, 3:45-61, 3:62-4:10.

g. *monitoring the computer network;*

2934. Bhattacharya discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example, “[t]he present invention relates generally to computer communications networks, and more particularly to regulating, ***monitoring and policing traffic on such computer communications networks.***” Bhattacharya at 1:15–18.

2935. Further examples include Bhattacharya at Abstract, 2:5–22, 3:25-45.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

2936. Bhattacharya discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. ***-The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes.*** Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2937. Further examples include Bhattacharya at 2:5–22, 3:14–18, 3:45-61, 5:54-61; Bhattacharya at Fig. 4.

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***prediction algorithm*

2938. Bhattacharya discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

QoS is a feature that allows dropping of packets on a selective basis to avoid or reduce congestion in the network. Two components of QoS are “classification” and “policing.” Packets are classified into different traffic classes according to policy set by the network administrator. For each class, a policing algorithm is used to measure the incoming traffic and compare that measure with policing parameters set by the network-administrator. As a result of policing, depending on the current traffic-rate for this class of traffic, a packet may be found “in profile” or “out of profile” by the policing algorithm. An out of profile packet is dropped or marked. ***Marking increases the probability of the packets being dropped later by another device that applies QoS to the packet. A packet that is dropped or marked by the policing algorithm is referred to as a “policed” packet.*** An in profile packet is forwarded without marking and is referred to as a packet “permitted” by the policing algorithm.

Bhattacharya at 2:5–22.

2939. Further examples include Bhattacharya at Abstract, 3:14–18.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

2940. Bhattacharya discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. ***Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node.*** The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2941. Further examples include Bhattacharya at 2:5–22.

- k. *dynamically modifying the assigned goal of the software agent by*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*replacing the assigned goal based on the optimal policy*

2942. Bhattacharya discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.”

’730 Pat. at Cl. 1. For example:

QoS is a feature that allows dropping of packets on a selective basis to avoid or reduce congestion in the network. Two components of QoS are “classification” and “policing.” ***Packets are classified into different traffic classes according to policy set by the network administrator. For each class, a policing algorithm is used to measure the incoming traffic and compare that measure with policing parameters set by the network-administrator. As a result of policing, depending on the current traffic-rate for this class of traffic, a packet may be found “in profile” or “out of profile” by the policing algorithm.*** An out of profile packet is dropped or marked. Marking increases the probability of the packets being dropped later by another device that applies QoS to the packet. A packet that is dropped or marked by the policing algorithm is referred to as a “policed” packet. An in profile packet is forwarded without marking and is referred to as a packet “permitted” by the policing algorithm.

Bhattacharya at 2:5–22.

2943. Further examples include Bhattacharya at 3:25-45, 3:45-61.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2944. Bhattacharya discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. ***An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node.*** Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. ***The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes.*** Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2945. Further examples include Bhattacharya at 3:25-45, 3:45-61.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

2946. Bhattacharya discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. ***Traffic policy parameters are established*** for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2947. Further examples include Bhattacharya at 2:5–22.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

2948. Bhattacharya discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the ***individual policer will export the traffic conditions at the respective node***. The ***other individual policers or a master policer will receive the exported information***. -The ***individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes***. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2949. Further examples include Bhattacharya at 3:25-45, 3:45-61, 3:62-4:10.

- b. *constructing a topological representation of the computer network from the information.*

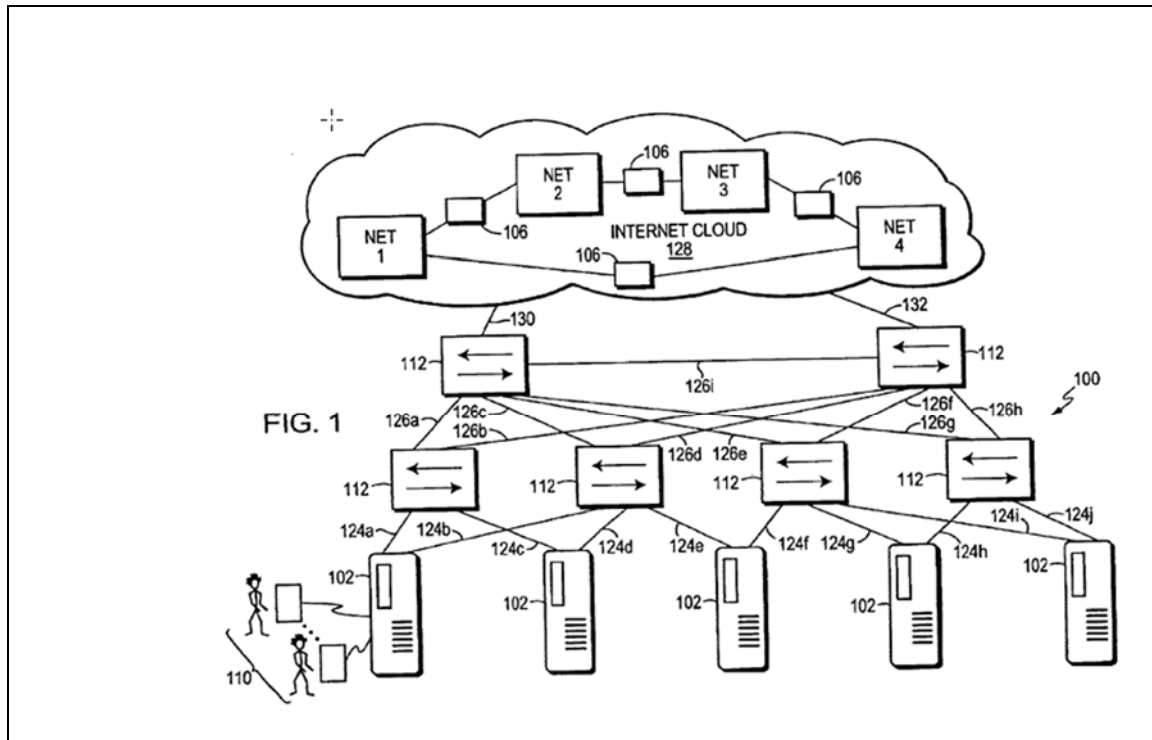
2950. Bhattacharya discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

FIG. 1 is a highly schematic illustration of a computer network **100**. The network **100** includes a plurality of servers **102** that are preferably configured as web-hosting servers. A number of users **110** are shown connecting to one of the host **102**, but typically there will be a number of users that are connected to each host. The network **100** further includes a plurality of intermediate network devices **112**, such as backbone routers, high-speed switches, etc. Each host **102** is preferably coupled to two or more of the devices **112** by respective trunks or links **124(a . . . j)**. Significantly, the network **100** is formed from a multitude of smaller networks, herein shown as NET 1-4. These smaller networks are interconnected together by routers **106**. These smaller included networks, considered all together are the Internet **128**. In a preferred embodiment, the devices **112** and the routers **106** may include similar designs. Links **126(a . . . i)** interconnect devices **112**, and links **130, 132** connect devices **112** to Internet **128**.

It should be understood that the configuration of network **100** is meant for illustrative purposes only, and that *the present invention will operate with other, possibly far more complex, network designs or topologies.*

Quality of service policing of individual switches and routers is well known in the field, and the present invention can be used advantageously with virtually any known policing system. Since such policers are so well known only a brief overview is provided herein. More fully developed details may be found in the U.S. patent application, assigned to the same entity as is this application, entitled METHOD AND APPARATUS FOR PERFORMING HIGH SPEED TRAFFIC SHAPING, Ser. No. 09/560,499 filed on Apr. 27, 2000, and incorporated herein by reference.

Bhattacharya at 4:56-22.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bhattacharya at Fig. 1

2951. Further examples include

4. **Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

2952. Bhattacharya discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. ***Leaky bucket algorithms may be used in some instances.***

Bhattacharya at Abstract.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

2953. Further examples include Bhattacharya at 2:5–22, 2:23–38, 3:14–18, 3:25–45, 3:45–61, 4:11–38.

2954. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

2955. Bhattacharya discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

***Various algorithms exist to perform policing.*** Each of these algorithms is implemented in a single logic-module called policer that performs the same computation for all packets belonging to a traffic class. Such a policer is referenced as a centralized policer, since the same module needs to perform the computation for all packets belonging to a traffic class. One such policing algorithm limits the total number bytes in all packets permitted in any arbitrary time-interval, T, to the value of (T\* contract\_rate+burst). Here and as defined below, “contract rate” is a policing policy parameter meaning information per unit time, and “burst” is another policing policy parameter meaning the maximum information permitted in excess of the rate. This can be implemented in a policer called the token-bucket policer, which performs the following computation for every packet in a traffic class.

Bhattacharya at 2:23–38.

2956. Further examples include Bhattacharya at 3:14–18, 3:25–45, 3:45–61.

2957. Moreover, I note the Dijkstra self-stabilization and similar numerical models were widely known at the time of the invention. A person of ordinary skill in the art would have known to include Dijkstra's algorithm in the model.

6. ***Claim 7***

- a. *A computer network, comprising:*

2958. Bhattacharya discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

A method for policing traffic on a ***computer communications network having a multitude of nodes interconnected by various communications media***. An individual policer is established at each node for monitoring



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. The other individual policers or a master policer will receive the exported information. -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2959. Further examples include Bhattacharya at 1:15–18, 3:25-45; Bhattacharya at Figs.

1, 2.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

2960. Bhattacharya discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

- c. *wherein the software agent has its own runtime environment*

2961. Bhattacharya discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- d. *is able to communicate with other software agents in the computer network*

2962. Bhattacharya discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

- e. *is capable of perceiving its own state; and*

2963. Bhattacharya discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

f. *is able to clone itself;*

2964. Bhattacharya discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

2965. Bhattacharya discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

QoS is a feature that allows dropping of packets on a selective basis to avoid or reduce congestion in the network. Two components of QoS are “classification” and “policing.” Packets are classified into different traffic classes according to policy set by the network administrator. For each class, ***a policing algorithm is used to measure the incoming traffic and compare that measure with policing parameters set by the network-administrator.*** As a result of policing, depending on the current traffic-rate for this class of traffic, a packet may be found “in profile” or “out of profile” by the policing algorithm. An out of profile packet is dropped or marked. Marking increases the probability of the packets being dropped later by another device that applies QoS to the packet. A packet that is dropped or marked by the policing algorithm is referred to as a “policed” packet. An in profile packet is forwarded without marking and is referred to as a packet “permitted” by the policing algorithm.

Bhattacharya at 2:5–22.

2966. Further examples include Bhattacharya at 3:45-61, 5:54-61; Bhattacharya at Fig. 4.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

2967. Bhattacharya discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

2968. Bhattacharya discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

2969. Bhattacharya discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

2970. Bhattacharya discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

2971. Bhattacharya discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**7. Claim 10**

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

2972. Bhattacharya discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

**8. Claim 11**

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

2973. Bhattacharya discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

A method for policing traffic on a computer communications network having a multitude of nodes interconnected by various communications media. An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. ***Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node.*** The other individual policers or a master policer will receive the exported information. ***-The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes.*** Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.

Bhattacharya at Abstract.

2974. Further examples include Bhattacharya at 1:15–18, 3:20-24, 3:25-45.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

2975. Bhattacharya discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

A method for policing traffic on a computer communications network ***having a multitude of nodes interconnected by various communications media.*** An individual policer is established at each node for monitoring and/or policing the traffic incoming to that node. Traffic policy parameters are established for traffic-classes and the policy is implemented at each individual policer. Thresholds may be established and when the thresholds are met or exceeded the individual policer will export the traffic conditions at the respective node. ***The other individual policers or a master policer will receive the exported information.*** -The individual policers police the traffic incoming to its associated node depending on the traffic condition information received from all the nodes. Several classes may be handled by each individual policer. Leaky bucket algorithms may be used in some instances.” Bhattacharya at Abstract.

2976. Further examples include Bhattacharya at 1:15–18, 2:5–22, 3:45-61.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

2977. Bhattacharya discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

Various algorithms exist to perform policing. Each of these algorithms is implemented in a single logic-module called policer that performs the same computation for all packets belonging to a traffic class. Such a policer is referenced as a centralized policer, since the same module needs to perform the computation for all packets belonging to a traffic class. One such policing algorithm limits the total number bytes in all packets permitted in any arbitrary time-interval, T, to the value of  $(T * \text{contract\_rate} + \text{burst})$ . Here and as defined below, “contract rate” is a policing policy parameter meaning information per unit time, and “burst” is another policing policy parameter meaning the maximum information permitted in excess of the rate. This can be implemented in a policer called the token-bucket policer, which performs the following computation for every packet in a traffic class.

Bhattacharya at 2:23–38.

2978. Further examples include

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

2979. Bhattacharya discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

Various algorithms exist to perform policing. Each of these algorithms is implemented in a single logic-module called policer that performs the same computation for all packets belonging to a traffic class. Such a policer is referenced as a centralized policer, since the same module needs to perform the computation for all packets belonging to a traffic class. ***One such policing algorithm limits the total number bytes in all packets permitted in any arbitrary time-interval, T, to the value of  $(T * \text{contract\_rate} + \text{burst})$ .*** Here and as defined below, “contract rate” is a policing policy parameter meaning information per unit time, and “burst” is another policing policy parameter meaning the maximum information permitted in excess of the rate. This can be implemented in a policer called the token-bucket policer, which performs the following computation for every packet in a traffic class.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bhattacharya at 2:23–38.

2980. Further examples include Bhattacharya at 4:11-38.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

2981. Bhattacharya discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

QoS is a feature that allows dropping of packets on a selective basis to avoid or reduce congestion in the network. Two components of QoS are “classification” and “policing.” Packets are classified into different traffic classes according to policy set by the network administrator. For each class, a policing algorithm is used to measure the incoming traffic and compare that measure with policing parameters set by the network-administrator. ***As a result of policing, depending on the current traffic-rate for this class of traffic, a packet may be found “in profile” or “out of profile” by the policing algorithm. An out of profile packet is dropped or marked.*** Marking increases the probability of the packets being dropped later by another device that applies QoS to the packet. A packet that is dropped or marked by the policing algorithm is referred to as a “policed” packet. An in profile packet is forwarded without marking and is referred to as a packet “permitted” by the policing algorithm.”

Bhattacharya at 2:5–22.

2982. Further examples include Bhattacharya at 3:25-45, 3:25-45, 5:34-53, 5:54-61; Bhattacharya at Fig. 4.

13. ***Claim 19***

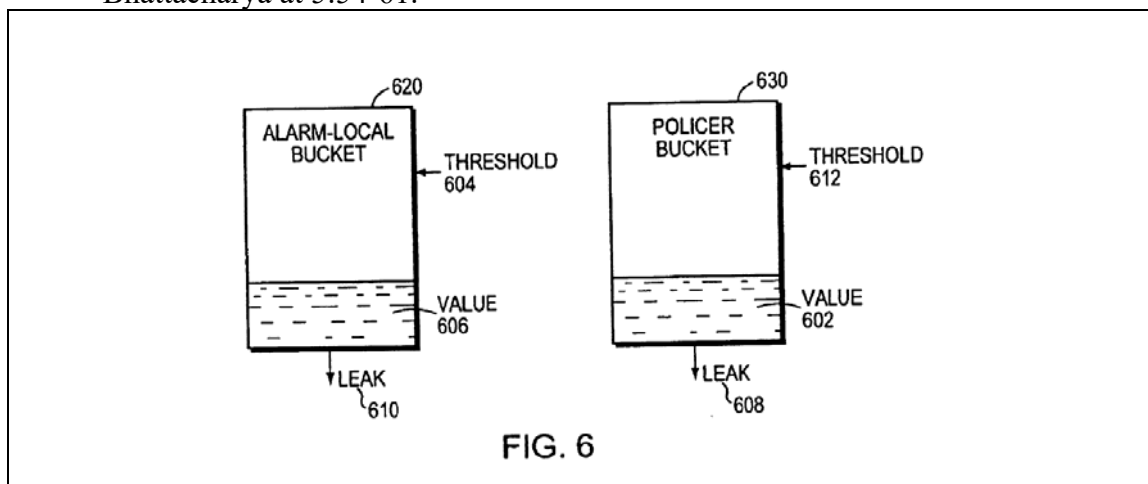
- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

2983. Bhattacharya discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

FIG. 6 shows diagrammatically the logic of a individual policer which implements local bucket based PUP transmission. In a preferred embodiment, ***the traffic policer algorithm may be implemented in hardware through a plurality of registers and combinational logic configured to produce sequential logic circuits and cooperating state machines.***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bhattacharya at 5:54-61.



Bhattacharya at Fig. 6.

2984. Further examples include

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

2985. Bhattacharya discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

The individual policer includes logic 226 and memory 228. The individual policer logic transmits packets, called the Policer Update Packet or PUP to other individual policers. ***The memory in the individual-policer, referenced as policer-memory, stores a record for each traffic class.***

Bhattacharya at 5:62–66.

2986. Bhattacharya then gives several examples of “variables [that] are store in each such record:”

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

| Policer<br>memory fields | Description  |
|--------------------------|--|
| contract_rate            | Parameter used to define policing policy for a traffic class in distributed token bucket algorithm here meaning information per unit time.                       |
| burst                    | Parameter used to define policing policy for a traffic class in distributed token bucket algorithm here meaning maximum information permitted in excess of rate. |
| byte_count_local         | Local state variable stored by an individual policer for a traffic class to implement the distributed token bucket algorithm                                     |
| bucket                   | Global state variable stored by an individual policer for a traffic class to implement the distributed token bucket algorithm                                    |
| last_update time         | Global state variable stored by an individual policer for a traffic class to implement the distributed token bucket algorithm                                    |
| byte_threshold           | Parameter used in one mechanism of triggering PUP transmission.  |
| time_threshold           | Parameter used in one mechanism of triggering PUP transmission.  |

Bhattacharya at 5:66–6:23.

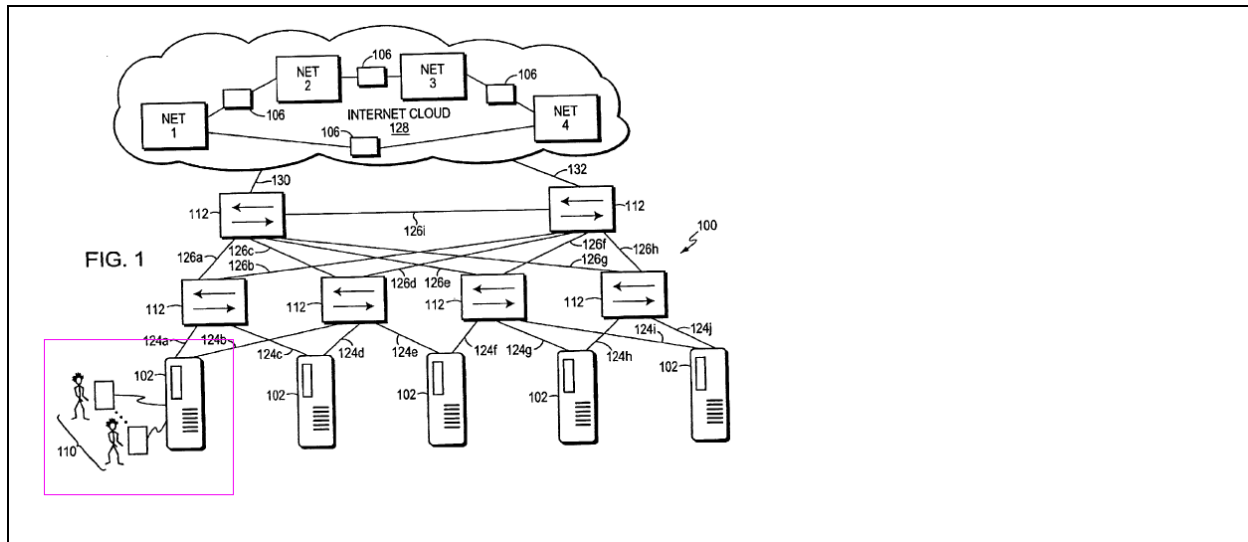
2987. Further examples include Bhattacharya at 3:25-45, 3:45-61.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

2988. Bhattacharya discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example, FIG. 1 is a highly schematic illustration of a computer network 100. The network 100 includes a plurality of servers 102 that are preferably ***configured as web-hosting servers***. ***A number of users 110*** are shown connecting to one of the host 102, but typically there will a number of users that are connected to each host.” Bhattacharya at 4:55-4:65.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bhattacharya at Fig. 1.

16. **Claim 24**

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

2989. Bhattacharya discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. **Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

2990. Bhattacharya discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

The present invention provides individual policers that monitor and police part of the traffic. In order to accomplish this policing by individual policers the contract rate and the burst for an entire traffic class is provided to the individual policers. The individual policers measure their parts of the traffic and export that information to all the other individual policers. There is a mechanism for receiving, totalizing and storing the individual policer information exchanged from all the individual policers. ***That total is then compared to the contract rate and burst for the entire traffic class and a policing decision is made by each individual policer for its part of the traffic class.*** In an embodiment, there is a master policer that receives all the information from all the individual policers and applies the contract rate

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

and the burst and makes policing decisions for the individual policers that is then sent back to the individual policers.

Bhattacharya at 3:45-61.

2991. Further examples include

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

2992. Bhattacharya discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

In a preferred embodiment, the policing decision for each packet is made using a modified version of the token bucket algorithm, and including thresholds wherein the information messages between individual policers occurs only when these thresholds are exceeded. Such thresholds may be on the data received and/or time from last up date or combinations of both. Moreover, it is possible to interchange the information between the individual policers via channels or a medium outside that of the network itself. For example, dial up telephone/fax lines, and the like. However, the same principle may be employed to other centralized policing algorithms.

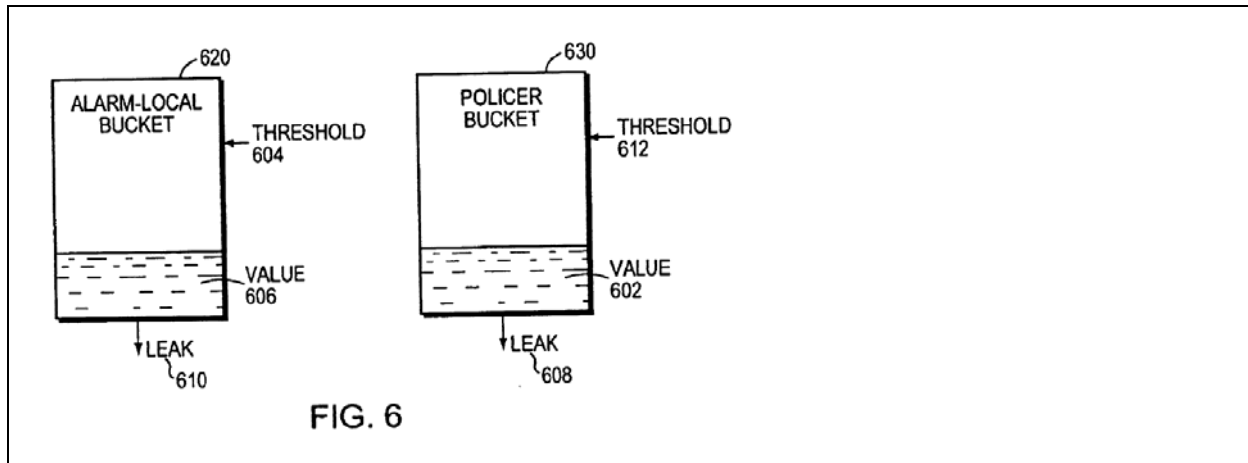
Bhattacharya at 4:20-32.

2993. Further examples include Bhattacharya at 3:45-61, 5:54-61.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

2994. Bhattacharya discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example, FIG. 6 shows diagrammatically the logic of a individual policer which implements local bucket based PUP transmission. In a preferred embodiment, *the traffic policer algorithm may be implemented in hardware through a plurality of registers and combinational logic configured to produce sequential logic circuits and cooperating state machines.*” Bhattacharya at 5:54-61.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Bhattacharya at Fig. 6.

2995. Further examples include Bhattacharya at 3:25-45, 5:34-53; Bhattacharya at Fig. 4.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

2996. Bhattacharya discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

2997. Bhattacharya discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

2998. Bhattacharya discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

2999. Bhattacharya discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy when it lacks an ability to perform the predefined task*

3000. Bhattacharya discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3001. Bhattacharya discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3002. Bhattacharya discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

i. *including predicting a failure of a network component based on a predictive algorithm;*

3003. Bhattacharya discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3004. Bhattacharya discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3005. Bhattacharya discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3006. Bhattacharya discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3007. Bhattacharya discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

3008. Bhattacharya discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3009. Bhattacharya discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3010. Bhattacharya discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3011. Bhattacharya discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3012. Bhattacharya discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

3013.

**X. Anticipation by and/or Obviousness in View of United States Patent No. 6,219,719, filed on 1/22/1997 and issued to Graf on 4/17/2001.**

3014. As explained in detail below and in the chart attached as Ex. A-23, Graf anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

3015. Graf discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” '730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter "SYSTEMWatch AI-L", comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3016. Further examples include Graf at 3:14–22, 4:6–42, 4:43–5:9; Graf at Fig. 1.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3017. Graf discloses this element of claim 1, "assigning a goal to a software [agent], wherein the software agent has its own runtime environment." '730 Pat. at Cl. 1. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter "SYSTEMWatch AI-L", comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3018. Further examples include Graf at 3:32–34, 3:44–48, 4:14–42, 4:43–5:8, 5:10–37, 6:28–7:6, 7:38–8:2, 8:36–43, 9:57–10:4, 10:25–39, 12:54–13:9, 15:48–61, 37:54–38:65; Graf at Figs. 1, 4.

- c. *is able to communicate with other software agents in the computer network;*

3019. Graf discloses this element of claim 1, "is able to communicate with other software agents in the computer network." '730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter "SYSTEMWatch AI-L", comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3020. Further examples include Graf at 4:6–42, 4:43–5:9, 10:25–48.

d. *is capable of perceiving its own state*

3021. Graf discloses this element of claim 1, "is capable of perceiving its own state." '730

Pat. at Cl. 1. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter "SYSTEMWatch AI-L", comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3022. Further examples include Graf at 3:14–22, 4:6–42, 4:43–5:9, 8:14–22.

e. *and is able to clone itself,*

3023. Graf discloses this element of claim 1, "and is able to clone itself." '730 Pat. at Cl.

1. For example:

In addition to ENTITYs and PROPERTYs, the database, 41, in SYSTEMWatch AI-L also has these additional features:

1. Host Information



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Each piece of data in database, 41, automatically has host information associated with it. Thus, as data is stored in the database, the database automatically associates the host from which the data originated from. This is because in SYSTEMWatch AI-L, data is “owned” by the host from where the data originated. Other hosts may request a copy of the data since SYSTEMWatch AI-L has communications capabilities. Some data may be stored in a central location (e.g.: a SYSTEMWatch AI-L console) if it is relevant to multiple computers. Because each piece of data has host information associated with it, a SYSTEMWatch AI-L console can consolidate data from multiple hosts.

## 2. Time Information

Each piece of data in database, 41, has a time field associated with it. The time field by default has the last time the data was updated, but SYSTEMWatch AI-L provides a mechanism of changing the time field so its possible to store some other time in the field.

## 3. Name

Each piece of data in database, 41, has a key field which is called the name field. A name field must be unique for a given ENTITY, PROPERTY, and host (the name of a computer). Thus, within an ENTITY and PROPERTY used for tracking computer processes, the name field might be the process id since process ids are unique on each computer, so by specifying the ENTITY name, PROPERTY name, and host name, the name field forms a unique key to locate the data.

## 4. Value

Of course, a database stores data. In SYSTEMWatch AI-L, the term value refers to the data stored in the database.

In one example, database, 41, is currently implemented as a relational database: One table is used for describing ENTITYs. This table is used to associate ENTYS with PROPERTYs. Another table is used for describing PROPERTYs. Finally, another table holds the information, which can be located by providing an ENTITY name, PROPERTY name, and the name field of the data. This table also contains the associated host and time information.

Graf at 6:28–7:6.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3024. Graf discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

“SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3025. Further examples include Graf at Abstract, 3:14–22, 3:44–48, 4:6–42, 3:32–34, 4:14–42, 4:43–5:8, 5:10–37, 6:28–7:6, 7:38–8:2, 8:36–43, 9:57–10:4, 10:25–39, 12:54–13:9, 15:48–61, 37:54–38:65; Graf at Figs. 1, 4.

g. *monitoring the computer network;*

3026. Graf discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3027. Further examples include Graf at 3:14–22, 4:6–42, 4:43–5:9, 37:54–38:65.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3028. Graf discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

After all the rules are declared, the expert system is in a state where it is ready to test rules. SYSTEMWatch AI-L forces the expert system

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

component of the core layer to run through its rules whenever the execRules function is called. As described later, the SYSTEMWatch AI-L client, 13, and SYSTEMWatch AI-L console, 21, each call a execRules function in their main loop. As shown in FIGS. 8a–8 b, in one embodiment, the expert system functions as follows:

First, if the rules have not been sorted, INQUERY 59, “Have the rules been sorted?”, the expert system reorders the rules by sorting them in specificity order, STEP 60. Rules are ranked in their order of specificity, with the most specific rules ordered before the least specific rules. Specificity is the total number of comparison operators (less than, less than or equal to, equal to, greater than, greater than or equal to, not equal to) and logical operators (AND, OR, NOT) contained within the boolean expression used as the test in the rules. For example, consider these boolean expressions:

| TABLE 2                 |             |
|-------------------------|-------------|
| Boolean Expression      | Specificity |
| A AND NOT B OR (C == D) | 4           |
| (A == B) && NOT C       | 3           |
| (A == B) && C           | 2           |
| A == B                  | 1           |
| TRUE                    | 0           |

If during the sorting, a group of rules has the same specificity, that group is sorted in declaration order, with the earlier declared rule ordered before a later declared rule. The reordering of the rules is only done once, during the first time the execRules function is called.

Graf at 8:53–9:18.

3029. Further examples include Graf at 9:25–48, 9:57–10:4, 25:16–27.

- i. *including predicting a failure of a network component based on a prediction algorithm*

3030. Graf discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3031. Further examples include Graf at 4:43–5:8 5:10–37, 8:36–43, 9:57–10:4, 12:54–13:9, 7:24–29, 14:13–39, 17:21–53, 21:52–66, 23:25–66, 27:1–20, 37:54–38:65, 40:34–47, 41:28–47.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3032. Graf discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

An Overview of the SYSTEM Watch AI-L Client

The task of the SYSTEMWatch AI-L client is to manage a computer and to provide notification of management actions to the SYSTEMWatch AI-L console. Before explaining how the SYSTEMWatch AI-L client operates, however, it is necessary to consider how the SYSTEMWatch AI-L client is organized. As previously mentioned, the SYSTEMWatch AI-L client is bifurcated into a core layer, **33**, which provides the SYSTEMWatch AI-L client with the underlying mechanism for detecting and responding to problems, and an application layer, **34**, which configures the SYSTEMWatch AI-L client to operate in a useful manner. The SYSTEMWatch AI-L client was designed this way because the nature of a particular computer's problem is not static. For example, problems may evolve as changes are made to the hardware and software of the computer, and if the computer is a multi-user computer, as users are added and removed from the system. As computer problems change, only the SYSTEMWatch AI-L client's application layer need be modified. As shown in FIG. 6, the core layer is composed of four elements: a database, **41**, an expert system, **40**, a language interpreter, **39**, and a communications mechanism, **42**. One example of a preferred embodiment of the application layer, **34**, is a series of programs written in a language which can be interpreted by the language interpreter of the core layer.

Graf at 5:10-37.

3033. Further examples include Graf at 8:36–43 9:57–10:4.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3034. Graf discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl. 1. For example: “What is needed is a tool which will automatically gather the necessary computer information to manage a group of computers, *detect problems based upon the gathered*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*information, inform the system administrator of detected problems, and automatically perform corrective actions to resolve detected problems.”* Graf at 3:6–11.

3035. Further examples include Graf at 4:43–5:8 5:10–37, 9:57–10:4, 17:21–53, 17:54–18:25, 20:21–60, 20:62–21–41, 37:54–38:65.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3036. Graf discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

3037. Further examples include Graf at 4:43–5:8, 5:10–37, 44:19–24, 8:36–43, 9:57–10:4, 12:54–13:9, 14:13–39, 17:21–53, 17:54, 18:25, 19:36–20:17, 20:21–60, 20:62–21–41, 21:52–66, 23:25–66, 27:1–20, 37:54–38:65, 40:34–47, 41:28–47.

**2. Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3038. Graf discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3039. Further examples include Graf at 3:32–3, 4:14–42, 4:43–5:8, 5:10–37, 6:28–7:6, 7:38–8:2, 8:36–43, 9:57–10:4, 10:25–39, 12:54–13:9, 15:48–61, 37:54–38:65, 3:44–48; Graf at Figs. 1, 4.

**3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3040. Graf discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3041. Moreover, Graf discloses this element of claim 3 for all the reasons explained regarding claim 1, element g.

3042. Further examples include Graf at 3:32–34, 3:44–48, 4:14–42, 4:43–5:8, 5:10–37, 6:28–7:6, 7:38–8:2, 8:36–43, 9:57–10:4, 10:25–39, 12:54–13:9, 15:48–61, 37:54–38:65; Graf at Figs. 1, 4.

- b. *constructing a topological representation of the computer network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

3043. Graf discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

3044. Further examples include Graf at 25:16–27, 11:24–31.

3045. Additionally, I note topological representations were widely known at the time of the invention. It would have been obvious to one ordinarily skilled in the art, and merely a design choice, to generate a topological representation.

**4. Claim 4**

a. *The method of claim 1, wherein the modeling uses a numerical method.*

3046. Graf discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

Note that if the system administrator wanted to add additional modules to detect, analyze, and respond to additional problems, he need only write a program in the high level language conforming to convention used in the other files in SYSTEMWatch AI-L and modify the application layer to read in his program(s) before the SYSTEMWatch AI-L client reads the ruleinit program.

Graf at 11:24–31.

3047. I also note that numerical models such as the Dijkstra self-stabilization algorithm were well-known at the time of the invention. A person of ordinary skill in the art would have seen it as obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm.

**5. Claim 6**

a. *The method of claim 4, wherein the numerical method comprises a*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Dijkstra Self Stabilization Algorithm.*

3048. Graf discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

Note that if the system administrator wanted to add additional modules to detect, analyze, and respond to additional problems, he need only write a program in the high level language conforming to convention used in the other files in SYSTEMWatch AI-L and modify the application layer to read in his program(s) before the SYSTEMWatch AI-L client reads the ruleinit program.

Graf at 11:24–31.

3049. I also note that numerical models such as the Dijkstra self-stabilization algorithm were well-known at the time of the invention. A person of ordinary skill in the art would have seen it as obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm.

6. ***Claim 7***

a. *A computer network, comprising:*

3050. Graf discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a SYSTEMWatch AI-L client which turns a computer into a managed computer, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

3051. Further examples include Graf at 3:14–22, 4:6–42, 4:43–5:9.

3052. Moreover, Graf discloses this element for at least all the reasons explained regarding claim 1, element a.

b. *a software agent having an assigned goal which is a programmatic*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*expression of a predefined task for the software agent embodied in hardware*

3053. Graf discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

3054. Graf discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

3055. Graf discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

3056. Graf discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

f. *is able to clone itself;*

3057. Graf discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3058. Graf discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

The system and method of this invention automatically manages a group of computers by automatically gathering data, storing the data, analyzing the stored data to identify specified conditions, and initiating automated actions to respond to the detected conditions. The invention, hereafter “SYSTEMWatch AI-L”, comprises a **SYSTEMWatch AI-L client which turns a computer into a managed computer**, a SYSTEMWatch AI-L console, which turns a computer into a monitoring computer, a SYSTEMWatch AI-L send facility, which allows a system administrator to send commands to various SYSTEMWatch AI-L clients through the SYSTEMWatch AI-L console, and a SYSTEMWatch AI-L report facility which allows a system administrator to query information collected and

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

processed by the SYSTEMWatch AI-L clients and SYSTEMWatch AI-L consoles.

Graf at Abstract.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3059. Graf discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3060. Graf discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

3061. Graf discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3062. Graf discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3063. Graf discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3064. Graf discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

3065. Graf discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

3066. Further examples include Graf at 4:43–5:9.

3067. Additionally, Graf discloses this element for at least the reasons explained regarding claim 1, element c above.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3068. Graf discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14–22.

3069. Further examples include Graf at 4:6–42.

3070. Additionally, Graf discloses this element for at least the reasons explained regarding claim 1, element c above.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3071. Graf discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component,” at least for all the reasons explained regarding claim 7, element g.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

3072. Graf discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.”

’730 Pat. at Cl. 17. For example:

13. actions:

When a problem is detected by SYSTEMWatch AI-L, the SYSTEMWatch AI-L client can be configured to automatically respond to the detected problem by initiating an automated action. However, if the SYSTEMWatch AI-L client is not configured to automatically respond to a problem, the system administrator can use the SYSTEMWatch AI-L console to command a particular SYSTEMWatch AI-L client to perform an action in response to a detected problem. The routines found in the actions program are the routines which are executed when the SYSTEMWatch AI-L client receives a command from the SYSTwhen a problemEMWatch AI-L console to initiate an action. These functions are generally front end functions which then call the appropriate (and related) routine described

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

ear[lier]. The actions program only contains these routines; it does not make any database nor expert system declarations.”

Graf at 41:28–47.

3073. Further examples include Graf at 3:14–22, 43:37–43, 43:44–44:17, 20:47–51.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3074. Graf discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

The following actions are available to respond to problems detected by the filesys program:

| TABLE 26      |  |
|---------------|--|
| Action        | Description  |
| kill          | Kills the specified process by sending the process the UNIX kill signal.   |
| stoptmp       | Stops the specified process for a specified period of time by first sending the process a UNIX STOP signal, and sending the process a UNIX CONTINUE signal after the specified period of time has elapsed.   |
| stopload      | Stops the specified process until the 1 minute system load average drops beneath a specified load by first sending the process a UNIX STOP signal, and when the system load drops to the specified limit, by then sending the process a UNIX CONTINUE signal.  |
| nice5         | Set the specified process' nice value to 5.  |
| nice10        | Set the specified process' nice value to 10.   |
| nice15        | Set the specified process' nice value to 15.   |
| nice20        | Set the specified process' nice value to 20.   |
| schedule10    | Reschedules a process so that it run approximately 10% of the time. Schedule10 queries the database periodically to ascertain what percentage of the CPU the specified process is consuming. If the process uses more than the goal percent CPU consumption, it is reniced such that it uses less CPU resources. If the process uses less than the goal percent CPU consumption, it is reniced so that it uses more CPU resources. This action only uses non-privileged calls to renice. |
| schedule25    | Similar to schedule10, except the percent CPU goal is 25% instead of 10%.  |
| schedule50    | Similar to schedule10, except the percent CPU goal is 50% instead of 10%.  |
| scheduleVIP10 | Similar to schedule10, except this action can utilize privileged calls to renice as well as the normal non-privileged calls to renice. Privileged nice calls are those nice values which cause the UNIX operating system to give a process more CPU time than normally allowed. These calls are privileged because only a process running with an effective user id of root (the UNIX “superuser”) may assign such a nice value to a process.  |
| scheduleVIP25 | Similar to schedule25, except this action can utilize privileged calls to renice as well as the normal non-privileged calls to renice.   |
| scheduleVIP50 | Similar to schedule50, except this action can utilize privileged calls to renice as well as the normal non-privileged calls to renice.   |

Graf at 33:6–47.

3075. Further examples include Graf at 40:34–47, 41:28–47, 41:47–66, 42:29–65.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3076. Graf discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

3077. An Overview of the SYSTEM Watch AI-L Client

The task of the SYSTEMWatch AI-L client is to manage a computer and to provide notification of management actions to the SYSTEMWatch AI-L console. Before explaining how the SYSTEMWatch AI-L client operates, however, it is necessary to consider how the SYSTEMWatch AI-L client is organized. As previously mentioned, the SYSTEMWatch AI-L client is bifurcated into a core layer, 33, which provides the SYSTEMWatch AI-L client with the underlying mechanism for detecting and responding to problems, and an application layer, 34, which configures the SYSTEMWatch AI-L client to operate in a useful manner. The SYSTEMWatch AI-L client was designed this way because the nature of a particular computer's problem is not static. For example, problems may evolve as changes are made to the hardware and software of the computer, and if the computer is a multi-user computer, as users are added and removed from the system. As computer problems change, only the SYSTEMWatch AI-L client's application layer need be modified. As shown in FIG. 6, the core layer is composed of four elements: a database, 41, an expert system, 40, a language interpreter, 39, and a communications mechanism, 42. One example of a preferred embodiment of the application layer, 34, is a series of programs written in a language which can be interpreted by the language interpreter of the core layer.

Graf at 5:10–37.

3078. Further examples include Graf at 47:13–18, 6:28–7:6.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3079. Graf discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

Core Layer Description—The Expert System

The second element of the core layer is an expert system, 40, which is used for problem detection and action initiation. The expert system, 40, is a forward chaining rule based expert system using a rule specificity algorithm. When SYSTEMWatch AI-L client, 13, is started, the expert

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

system contains no rules. Rules are declared and incorporated into the core layer. Rules support both the IF-THEN rules as well as IF-THEN-ELSE rules. The rules used in SYSTEMWatch AI-L permit assignments and function calls within the condition of the rule. Additionally, SYSTEMWatch AI-L expert system, **40**, also has the following features:

- a. Rules can declare variables. All variables declared within a rule are static variables.
- b. Rules can have an initialization section. The initialization section contains actions which must be performed only once, and before the rule is ever tested. It can, for example, contain a state declaration and an interval declaration (states and intervals are described below). It may contain variable declarations for variables used by the rules, and it may contain code to do a variety of actions.
- c. Rules can have, for instance, an INTERVAL and a LASTCHECK time. In accordance with the principles of the present invention, in order for a rule to be eligible for testing by the expert system, at the time of testing the clock time must be equal to or greater than the LASTCHECK time plus the INTERVAL time. The LASTCHECK time for each rule is set to the clock time whenever a rule is actually tested. This way, the INTERVAL specifies the minimum amount of time which must elapse since the last time a rule was checked before the rule becomes eligible for testing again.

Graf at 7:38–8:2.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3080. Graf discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3081. Graf discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

3082. Graf discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy,” at least for all the reasons explained regarding claim 1, elements, h, i, j, and k.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3083. Graf discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

13. actions:

When a problem is detected by SYSTEMWatch AI-L, the SYSTEMWatch AI-L client can be configured to automatically respond to the detected problem by initiating an automated action. However, if the SYSTEMWatch AI-L client is not configured to automatically respond to a problem, the system administrator can use the SYSTEMWatch AI-L console to command a particular SYSTEMWatch AI-L client to perform an action in response to a detected problem. The routines found in the actions program are the routines which are executed when the SYSTEMWatch AI-L client receives a command from the SYSTEMWatch AI-L console to initiate an action. These functions are generally front end functions which then call the appropriate (and related) routine described ear[lier]. The actions program only contains these routines; it does not make any database nor expert system declarations.

Graf at 41:28–47

3084. Further examples include Graf at 3:14–22, 43:37–43, 20:47–51.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3085. Graf discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

The shortcomings of the prior art are overcome and additional advantages are provided in accordance with the principles of the present invention



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

through the provision of SYSTEMWatch AI-L, which automatically manages at least one computer by automatically gathering computer information, storing the gathered information, analyzing the stored information to identify specific computer conditions, and performing automatic actions based on the identified computer conditions.

Graf at 3:14-22.

3086. Further examples include all reasons explained regarding claims 1 and 7, element

a.

b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3087. Graf discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

c. *is able to communicate with other software agents in the computer network*

3088. Graf discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

d. *is capable of perceiving its own state; and*

3089. Graf discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

e. *is able to clone itself;*

3090. Graf discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3091. Graf discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

g. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

3092. Graf discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3093. Graf discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3094. Graf discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3095. Graf discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3096. Graf discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*assigned goal of the agent is expressed as a policy.*

3097. Graf discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. **Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3098. Graf discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

3099. Graf discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. **Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3100. Graf discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*characteristic.*

3101. Graf discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3102. Graf discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3103. Graf discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

3104. I expect to testify that Graf anticipates and/or renders obvious each Asserted Claim of the '730 Patent. A claim chart illustrating that Graf discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-23.

**Y. Anticipation by and/or Obviousness in View of United States Patent No. 6,243,667, filed 5/28/1996 and issued to Kerr, et al. on 6/5/2001.**

3105. As explained in detail below and in the chart attached as Ex. A-24, Kerr anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.

1. ***Claim 1***

- a. *A method of managing a computer network, comprising:*

3106. Kerr discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” '730 Pat. at Cl. 1. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

“The invention provides a *method* and system for switching in networks responsive to message flow patterns.”

Kerr at 1:48–49.

3107. Further examples include Kerr at 2:14–15, 2:21–22, Cl. 16, Fig. 2, Fig. 5.

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3108. Kerr discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

***A message "flow" is defined to comprise a set of packets to be transmitted between a particular source and a particular destination.*** When routers in a network identify a new message flow, they determine the proper processing for packets in that message flow and cache that information for that message flow. Thereafter, when routers in a network identify a packet which is part of that message flow, they process that packet according to the proper processing for packets in that message flow.

Kerr at 1:49–58.

3109. Further examples include Kerr at 2:31–33, 2:56–3:2, 8:50–65.

- c. *is able to communicate with other software agents in the computer network;*

3110. Kerr discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

***A message "flow" is defined to comprise a set of packets to be transmitted between a particular source and a particular destination.*** When routers in a network identify a new message flow, they determine the proper processing for packets in that message flow and cache that information for that message flow. Thereafter, when routers in a network identify a packet which is part of that message flow, they process that packet according to the proper processing for packets in that message flow.

Kerr at 1:49–58.

3111. Further examples include Kerr at 2:56–3:2, 8:50–65, Fig. 1.

- d. *is capable of perceiving its own state*

3112. Kerr discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

At a step 222, the routing device 140 identifies a message flow 160 for the packet 150. In a preferred embodiment, the routing device 140 examines a header for the packet 150 and identifies the IP address for the Source device

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

120, the IP address for the destination device 130, and the protocol type for the packet 150. The routing device 140 determines the port number for the source device 120 and the port number for the destination device 130 responsive to the protocol type. Responsive to this set of information, the routing device 140 determines a flow key 310 (described with reference to FIG. 3) for the message flow 160. At a step 223, the routing device 140 performs a lookup in a flow cache for the identified message flow 160. If the lookup is unsuccessful, the identified message flow 160 is a “new” message flow 160, and the routing device 140 continues with the step 224. If the lookup is successful, the identified message flow 160 is an “old” message flow 160, and the routing device 140 continues with the step 225. In a preferred embodiment, the routing device 140 determines a hash table key responsive to the flow key 310. This aspect of the step 223 is described in further detail with regard to FIG. 3. At a step 224, the routing device 140 builds a new entry in the flow cache. The routing device 140 determines proper treatment of packets 150 in the message flow 160 and enters information regarding Such proper treatment in a data Structure pointed to by the new entry in the flow cache. In a preferred embodiment, the routing device 140 determines the proper treatment by performing a lookup in an IP address cache as shown in FIG. 4.

...

Thereafter, the routing device 140 proceeds with the step 225, using the information from the new entry in the flow cache, just as if the identified message flow 160 were an “old” message flow 160 and the lookup in a flow cache had been Successful. At a step 225, the routing device 140 retrieves routing information from the entry in the flow cache for the identified message flow 160.

...

At a step 226, the routing device 140 routes the packet 150 responsive to the routing information retrieved at the Step 225.

Kerr at 3:57–4:63. *See also* Fig. 2A, Fig. 3, Fig. 4.

e. *and is able to clone itself,*

3113. Kerr discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl.

1. For example:

“1. A method for routing messages in a data network wherein a set of packets is isolated for Specialized policy treatment by a plurality of routing devices in the data network, the method comprising the Steps of: . . . .”

Kerr at Cl. 1.

3114. Further examples include Kerr at Cl. 11, Cl. 16.

f. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent; and*

3115. Kerr discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

g. *monitoring the computer network;*

3116. Kerr discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

3117. Further examples include Kerr at 3:40–45, 5:37–48, 6:32–41, 8:50–9:10, Cl. 16, Fig. 3.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3118. Kerr discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters

Kerr at 1:62–2:6.

3119. Further examples include Kerr at 8:50–9:10.

- i. *including predicting a failure of a network component based on a prediction algorithm*

3120. Kerr discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

3121. Further examples include Kerr at 8:50–9:10.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3122. Kerr discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

3123. Further examples include Kerr at 8:50–9:10.

- k. *dynamically modifying the assigned goal of the software agent by*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*replacing the assigned goal based on the optimal policy*

3124. Kerr discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.” ’730 Pat. at Cl.

1. For example:

For a second example, the routing device 140 is able to monitor usage information regarding relative use of network resources, and to give priority to those message flows 160 which use relatively fewer network resources. This can occur when a first message flow 160 is using a relatively low-bandwidth transmission channel (such as a 28.8 kilobits per second modem transmission channel) and when a second message flow 160 is using a relatively high-bandwidth transmission channel (such as a T-1 transmission line).

Kerr at 5:37–48.

3125. Further examples include Kerr at 3:40–45, 4:12–33, 4:64–5:10, 8:60–9:10, Fig. 3, Fig. 4.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3126. Kerr discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

[T]he method for routing in networks responsive to message flow patterns comprises two parts. In a first part, the routing device 140 builds and uses a flow cache described in further detail with regard to FIG. 3), in which routing information to be used for packets 150 in each particular message flow 160 is recorded and from which such routing information is retrieved for use.

Kerr at 3:40–45.

3127. Further examples include Kerr at 1:62–2:6, 6:32–41, 6:50–64, 8:50–9:10, Fig. 3.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3128. Kerr discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Because the routing device 140 processes each packet 150 in the message flow 160 responsive to the entry for the message flow 160 in the flow cache, the routing device 140 is able to implement administrative policies which are designated for each message flow 160 rather than for each packet 150. For example, the routing device 140 is able to reserve specific amounts of bandwidth for particular message flows 160 and to queue packets 150 for transmission responsive to the bandwidth reserved for their particular message flows 160.

Kerr at 5:17–25.

3. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3129. Kerr discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

[I]nformation about message flow patterns is collected, responsive to identified message flows and their packets. The collected information is reported to devices on the network. The collected information is used for a variety of purposes, including: to diagnose actual or potential network problems, to determine patterns of usage by date and time or by location, to determine which services and which users use a relatively larger or smaller amount of network resources, to determine which services are accessed by particular users, to determine which users access particular services, or to determine usage which falls within selected parameters.

Kerr at 1:62–2:6.

3130. Further examples include Kerr at 3:40–45, 5:37–48, 6:32–41, 8:50–9:10, Cl. 16, Fig. 3.

- b. *constructing a topological representation of the computer network from the information.*

3131. Kerr discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

Interested parties may determine patterns of usage of the network by date and time or by location. For example, the report may comprise information about which users or which services on the network are making relatively heavy use of resources. In a preferred embodiment, usage of the network 100 is displayed in a graphical form which shows use of the network 100 in a false-color map, so that network administrators and other interested parties may rapidly determine which services, which users, and which

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

communication links are relatively loaded or relatively unloaded with demand.

Kerr at 9:12–22.

**4. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3132. Kerr discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

For a second example, the routing device 140 is able to monitor usage information regarding relative use of network resources, and to give priority to those message flows 160 which use relatively fewer network resources. This can occur when a first message flow 160 is using a relatively low-bandwidth transmission channel (such as a 28.8 kilobits per second modem transmission channel) and when a second message flow 160 is using a relatively high-bandwidth transmission channel (such as a T-1 transmission line).”

Kerr at 5:37–48.

3133. Further, numerical models such as the Dijkstra self-stabilization algorithm were wellknown at the time of the invention. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm in view of a person of ordinary skill in the art.

**5. Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3134. Kerr discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

For a second example, the routing device 140 is able to monitor usage information regarding relative use of network resources, and to give priority to those message flows 160 which use relatively fewer network resources. This can occur when a first message flow 160 is using a relatively low-bandwidth transmission channel (such as a 28.8 kilobits per second modem transmission channel) and when a second message flow 160 is using a relatively high-bandwidth transmission channel (such as a T-1 transmission line).

Kerr at 5:37–48.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3135. Further, numerical models such as the Dijkstra self-stabilization algorithm were wellknown at the time of the invention. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm in view of a person of ordinary skill in the art.

6. *Claim 7*

a. *A computer network, comprising:*

3136. Kerr discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

“The invention provides a method and system for switching in networks responsive to message flow patterns.”

Kerr at 1:48–49.

3137. Further examples include Kerr at 2:31–33.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3138. Kerr discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

3139. Kerr discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

3140. Kerr discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

3141. Kerr discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

f. *is able to clone itself;*

3142. Kerr discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3143. Kerr discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

[T]he method for routing in networks responsive to message flow patterns comprises two parts. In a first part, the routing device 140 builds and uses a flow cache described in further detail with regard to FIG. 3), in which routing information to be used for packets 150 in each particular message flow 160 is recorded and from which such routing information is retrieved for use.

Kerr at 3:40–45.

3144. Further examples include Kerr at 2:31–33, 4:12–33, 4:64–5:10, Fig. 3, Fig. 4.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3145. Kerr discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3146. Kerr discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the prediction;*

3147. Kerr discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3148. Kerr discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3149. Kerr discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3150. Kerr discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

3151. Kerr discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

At a step 532, the reporting device 540 builds a report about a condition of the network 100, responsive to information about message flows 160. At a step 533, the reporting device 540 displays or transmits that report about the condition of the network 100 to interested parties.

Kerr at 8:60–65.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3152. Kerr discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12.

For example:

At a step 532, the reporting device 540 builds a report about a condition of the network 100, responsive to information about message flows 160. At a step 533, the reporting device 540 displays or transmits that report about the condition of the network 100 to interested parties.

Kerr at 8:60–65.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3153. Kerr discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

At a step 222, the routing device 140 identifies a message flow 160 for the packet 150. In a preferred embodiment, the routing device 140 examines a header for the packet 150 and identifies the IP address for the source device 120, the IP address for the destination device 130, and the protocol type for the packet 150. The routing device 140 determines the port number for the source device 120 and the port number for the destination device 130 responsive to the protocol type. Responsive to this set of information, the routing device 140 determines a flow key 310 (described with reference to FIG. 3) for the message flow 160.

Kerr at 3:57–67.

3154. Further examples include Kerr at 7:59–61, Fig. 3.

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

3155. Kerr discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.”

’730 Pat. at Cl. 17. For example:

“[T]he method for routing in networks responsive to message flow patterns comprises two parts. In a first part, the routing device 140 builds and uses a flow cache described in further detail with regard to FIG. 3), in which routing information to be used for packets 150 in each particular message flow 160 is recorded and from which such routing information is retrieved for use.”

Kerr at 3:40–45.

3156. Further examples include Kerr at 4:12–33, 4:64–5:10, Fig. 3, Fig. 4.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3157. Kerr discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

For example:

At a step 241, the routing device 140 examines each entry in the flow cache and compares a current time with a last time a packet 150 was routed using that particular entry. If the difference exceeds a first selected timeout, the message flow 160 represented by that entry is considered to have expired due to nonuse and thus to no longer be valid. In a preferred embodiment, the routing device 140 also examines the entry in the flow cache and compares a current time with a first time a packet 150 was routed using that particular entry. If the difference exceeds a second selected timeout, the message flow 160 represented by that entry is considered to have expired due to age and thus to no longer be valid. The second selected timeout is preferably about one minute. Expiring message flows 160 due to age artificially requires that a new message flow 160 must be created for the next packet 150 in the same communication session represented by the old message flow 160 which was expired. However, it is considered preferable to do so because it allows information to be collected and reported about message flows 60 without having to wait for those message flows 160 to expire from nonuse. For example, a multiple broadcast communication session could reasonably last well beyond the time message flows 160 are expired for age, and if not so expired would mean that information about network usage would not account for significant network usage. In a preferred embodiment, the routing device 140 also examines the entry in the flow cache and determines if the "next hop" information has changed. If so, the message flow 160 is expired due to changed conditions. Other changed conditions which might cause a message flow 160 to be expired



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

include changes in access control lists or other changes which might affect the proper treatment of packets 150 in the message flow 160. The routing device 140 also expires entries in the flow cache on a least-recently-used basis if the flow cache becomes too full. If the message flow 160 is still valid, the routing device 140 continues with the next entry in the flow cache until all entries have been examined. If the message flow 160 is no longer valid, the routing device 140 continues with the step 242.

Kerr at 5:52– 6:24.

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3158. Kerr discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

At a step 241, the routing device 140 examines each entry in the flow cache and compares a current time with a last time a packet 150 was routed using that particular entry. If the difference exceeds a first selected timeout, the message flow 160 represented by that entry is considered to have expired due to nonuse and thus to no longer be valid. In a preferred embodiment, the routing device 140 also examines the entry in the flow cache and compares a current time with a first time a packet 150 was routed using that particular entry. If the difference exceeds a second selected timeout, the message flow 160 represented by that entry is considered to have expired due to age and thus to no longer be valid. The second selected timeout is preferably about one minute. Expiring message flows 160 due to age artificially requires that a new message flow 160 must be created for the next packet 150 in the same communication session represented by the old message flow 160 which was expired. However, it is considered preferable to do so because it allows information to be collected and reported about message flows 60 without having to wait for those message flows 160 to expire from nonuse. For example, a multiple broadcast communication session could reasonably last well beyond the time message flows 160 are expired for age, and if not so expired would mean that information about network usage would not account for significant network usage. In a preferred embodiment, the routing device 140 also examines the entry in the flow cache and determines if the "next hop" information has changed. If so, the message flow 160 is expired due to changed conditions. Other changed conditions which might cause a message flow 160 to be expired include changes in access control lists or other changes which might affect the proper treatment of packets 150 in the message flow 160. The routing device 140 also expires entries in the flow cache on a least-recently-used basis if the flow cache becomes too full. If the message flow 160 is still valid, the routing device 140 continues with the next entry in the flow cache until all entries have been examined. If the message flow 160 is no longer valid, the routing device 140 continues with the step 242.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Kerr at 5:52– 6:24.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3159. Kerr discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

At a step 224, the routing device 140 builds a new entry in the flow cache. The routing device 140 determines proper treatment of packets 150 in the message flow 160 and enters information regarding such proper treatment in a data structure pointed to by the new entry in the flow cache. In a preferred embodiment, the routing device 140 determines the proper treatment by performing a lookup in an IP address cache as shown in FIG. 4. In a preferred embodiment, the proper treatment of packets 150 in the message flow 160 includes treatment with regard to switching (thus, the routing device 140 determines an output port for switching packets 150 in the message flow 160), with regard to access control (thus, the routing device 140 determines whether packets 150 in the message flow 160 meet the requirements of access control, as defined by access control lists in force at the routing device 140), with regard to accounting (thus, the routing device 140 creates an accounting record for the message flow 160), with regard to encryption (thus, the routing device 140 determines encryption treatment for packets 150 in the message flow 160), and any special treatment for packets 150 in the message flow 160.

Kerr at 4:13–33.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3160. Kerr discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

Interested parties may determine patterns of usage of the network by date and time or by location. For example, the report may comprise information about which users or which services on the network are making relatively heavy use of resources. In a preferred embodiment, usage of the network 100 is displayed in a graphical form which shows use of the network 100 in a false-color map, so that network administrators and other interested parties may rapidly determine which services, which users, and which communication links are relatively loaded or relatively unloaded with demand.

Kerr at 9:12–22.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3161. Kerr discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3162. Kerr discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl.

26. For example:

Because the routing device 140 processes each packet 150 in the message flow 160 responsive to the entry for the message flow 160 in the flow cache, the routing device 140 is able to implement administrative policies which are designated for each message flow 160 rather than for each packet 150. For example, the routing device 140 is able to reserve specific amounts of bandwidth for particular message flows 160 and to queue packets 150 for transmission responsive to the bandwidth reserved for their particular message flows 160.

Kerr at 5:17–25.

3163. Further examples include Kerr at 8:60–9:10, and all the reasons explained above regarding claims 1.7, 1.8, 1.9, and 1.10.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3164. Kerr discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

[T]he method for routing in networks responsive to message flow patterns comprises two parts. In a first part, the routing device 140 builds and uses a flow cache described in further detail with regard to FIG. 3), in which routing information to be used for packets 150 in each particular message

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

flow 160 is recorded and from which such routing information is retrieved for use.

Kerr at 3:40–45.

3165. Further examples include Kerr at 1:62–2:6, 6:32–41, 8:50–9:10, Fig. 3.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3166. Kerr discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

The invention provides a method and system for switching in networks responsive to message flow patterns.

Kerr at 1:48–49. See also Kerr at 2:31–33.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3167. Kerr discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

3168. Kerr discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

3169. Kerr discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

3170. Kerr discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3171. Kerr discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3172. Kerr discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3173. Kerr discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3174. Kerr discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3175. Kerr discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3176. Kerr discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3177. Kerr discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3178. Kerr discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

3179. Kerr discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3180. Kerr discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3181. Kerr discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3182. Kerr discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3183. Kerr discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

3184. I expect to testify that Kerr anticipates and/or renders obvious each Asserted Claim of the '730 Patent. A claim chart illustrating that Kerr discloses and/or renders obvious each and every limitation of those claims is included as Ex. A-24.

**Z. Anticipation by and/or Obviousness in View of Cisco NetFlow, marketed by Cisco by 1995 at the latest.**

3185. As explained in detail below and in the chart attached as Ex. A-25, Cisco NetFlow anticipates and renders obvious the claims of the '730 Patent at least under the apparent application of the claims in NetFuel's infringement contentions.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3186. Cisco NetFlow discloses the preamble of claim 1, “[a] method of managing a computer network, comprising.” ’730 Pat. at Cl. 1. For example:

NetFlow is a Cisco IOS application that provides statistics on packets flowing through the routing devices in the network. It is emerging as a primary network accounting and security technology.

NetFlow Configuration Guide at 1.

3187. Further examples include NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1).

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3188. Cisco NetFlow discloses this element of claim 1, “assigning a goal to a software [agent], wherein the software agent has its own runtime environment.” ’730 Pat. at Cl. 1. For example:

NetFlow is available on all router platforms from the 2600 series upwards from the 12.0 software release onwards. It was first introduced in 11.1CC on the 7200 and 7500 platforms.

ISP Essentials at 44.

3189. Further examples include NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1), NetFlow Solutions Guide at § NetFlow Infrastructure, NetFlow Solutions Guide at § Why Deploy NetFlow?, NetFlow Configuration Guide at 2, NetFlow Solutions Guide at § Capturing NetFlow Data, Cyber Ops at 78.

c. *is able to communicate with other software agents in the computer network;*

3190. Cisco NetFlow discloses this element of claim 1, “is able to communicate with other software agents in the computer network.” ’730 Pat. at Cl. 1. For example:

The NetFlow MIB feature allows NetFlow statistics and other NetFlow data for the managed devices on your system to be retrieved by SNMP. You can specify retrieval of NetFlow information from a managed device (for example, a router) either by entering commands on that managed device or by entering SNMP commands from the NMS workstation to configure the router via the MIB. If the NetFlow information is configured from the NMS



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

workstation, no access to the router is required and all configuration can be performed via SNMP. The NetFlow MIB request for information is sent from an NMS workstation via SNMP to the router and is retrieved from the router. This information can then be stored or viewed, thus allowing NetFlow information to be easily accessed and transported across a multi-vendor programming environment.

NetFlow Configuration Guide at 198.

3191. Further examples include NetFlow Configuration Guide at 221, NetFlow Configuration Guide at 218.

d. *is capable of perceiving its own state*

3192. Cisco NetFlow discloses this element of claim 1, “is capable of perceiving its own state.” ’730 Pat. at Cl. 1. For example:

An additional benefit of the NetFlow MIB and Top Talkers feature is that it can be configured for a router either by entering CLI commands or by entering SNMP commands on a network management system (NMS) workstation. The SNMP commands are sent to the router and processed by a MIB. You do not have to be connected to the router console to extract the list of top talkers information if an NMS workstation is configured to communicate using SNMP to your network device.

NetFlow Configuration Guide at 221.

3193. Further examples include NetFlow Configuration Guide at 198, NetFlow Configuration Guide at 218, NetFlow Configuration Guide at 294, NetFlow Configuration Guide at 214.

e. *and is able to clone itself,*

3194. Cisco NetFlow discloses this element of claim 1, “and is able to clone itself.” ’730 Pat. at Cl. 1. For example:

NMS --network management system. A system responsible for managing at least part of a network. An NMS is generally a reasonably powerful and well-equipped computer, such as an engineering workstation. NMSs communicate with agents to help keep track of network statistics and resources.

NetFlow Configuration Guide at 218.

3195. Further examples include NetFlow Configuration Guide at 198, NetFlow Configuration Guide at 221.

f. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent; and*

3196. Cisco NetFlow discloses this element of claim 1, “and wherein the goal is a programmatic expression of a predefined task for the software agent.” ’730 Pat. at Cl. 1. For example:

User monitoring and profiling—NetFlow data enables you to gain detailed understanding of customer/user usage of network and application resources. This information may then be used to efficiently plan and allocate access, backbone, and application resources as well as to detect and resolve potential security and policy violations.

NetFlow Solutions Guide at § Why Deploy NetFlow?

3197. Further examples include NetFlow Solutions Guide at § Capturing NetFlow Data, ISP Essentials at 44, NetFlow Configuration Guide at 2, Cyber Ops at 78.

g. *monitoring the computer network;*

3198. Cisco NetFlow discloses this element of claim 1, “monitoring the computer network.” ’730 Pat. at Cl. 1. For example:

Network monitoring—NetFlow data enables extensive near real-time network monitoring capabilities. You can use NetFlow flow data analysis to display traffic patterns associated with individual routing devices and switches as well as on a networkwide basis (providing aggregate traffic or application-based views) to provide proactive problem detection, efficient troubleshooting, and rapid problem resolution.

NetFlow Solutions Guide at § Why Deploy NetFlow?

3199. Further examples include NetFlow Solutions Guide at § Capturing NetFlow Data, ISP Essentials at 44, NetFlow Configuration Guide at 2, Cyber Ops at 78.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3200. Cisco NetFlow discloses this element of claim 1, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network.” ’730 Pat. at Cl. 1. For example:

In this example, the first packet is being policy routed via route map test sequence 10. The subsequent packets of the same flow always take the same route map test sequence 10, not route map test sequence 20, because they all match or pass access list 1 check. Policy routing can be flow accelerated by bypassing the ACL check. The NetFlow cache entry size will be slightly

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

extended (increased from 64 bytes to 96 bytes for every single flow) to reserve extra space per flow for state information for other features. Features such as setting policy routing extra proprietary fields, setting packet precedence, and setting the next-hop address will be stored in the state information per flow. The flow state information is not visible and will not appear using the show ip cache verbose flow command.

NetFlow Solutions Guide at § NPR Benefits.

3201. Further examples include NetFlow Configuration Guide at 160.

- i. *including predicting a failure of a network component based on a prediction algorithm*

3202. Cisco NetFlow discloses this element of claim 1, “including predicting a failure of a network component based on a prediction algorithm.” ’730 Pat. at Cl. 1. For example:

Network Planning NetFlow can capture data over a long period of time, which enables you to track and anticipate network growth and plan upgrades. NetFlow service data can be used to optimize network planning, which includes peering, backbone upgrade planning, and routing policy planning. It also enables you to minimize the total cost of network operations while maximizing network performance, capacity, and reliability. NetFlow detects unwanted WAN traffic, validates bandwidth and quality of service (QoS) usage, and enables the analysis of new network applications. NetFlow offers valuable information that you can use to reduce the cost of operating the network.

NetFlow Configuration Guide at 2.

3203. Further examples include NetFlow Configuration Guide at 160.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3204. Cisco NetFlow discloses this element of claim 1, “wherein said modeling comprises determining appropriate policy based on the prediction.” ’730 Pat. at Cl. 1. For example:

Network Planning NetFlow can capture data over a long period of time, which enables you to track and anticipate network growth and plan upgrades. NetFlow service data can be used to optimize network planning, which includes peering, backbone upgrade planning, and routing policy planning. It also enables you to minimize the total cost of network operations while maximizing network performance, capacity, and reliability. NetFlow detects unwanted WAN traffic, validates bandwidth and quality of service (QoS) usage, and enables the analysis of new network applications. NetFlow offers valuable information that you can use to reduce the cost of operating the network.

NetFlow Configuration Guide at 2.

3205. Further examples include NetFlow Configuration Guide at 160.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3206. Cisco NetFlow discloses this element of claim 1, “dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy.”

’730 Pat. at Cl. 1. For example:

Network Planning NetFlow can capture data over a long period of time, which enables you to track and anticipate network growth and plan upgrades. NetFlow service data can be used to optimize network planning, which includes peering, backbone upgrade planning, and routing policy planning. It also enables you to minimize the total cost of network operations while maximizing network performance, capacity, and reliability. NetFlow detects unwanted WAN traffic, validates bandwidth and quality of service (QoS) usage, and enables the analysis of new network applications. NetFlow offers valuable information that you can use to reduce the cost of operating the network.

NetFlow Configuration Guide at 2.

3207. Further examples include NetFlow Configuration Guide at 160.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3208. Cisco NetFlow discloses this element of claim 1, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.” ’730 Pat. at Cl. 1. For example:

NetFlow can capture data over a long period of time, which enables you to track and anticipate network growth and plan upgrades. NetFlow service data can be used to optimize network planning, which includes peering, backbone upgrade planning, and routing policy planning. It also enables you to minimize the total cost of network operations while maximizing network performance, capacity, and reliability. NetFlow detects unwanted WAN traffic, validates bandwidth and quality of service (QoS) usage, and enables the analysis of new network applications. NetFlow offers valuable information that you can use to reduce the cost of operating the network.

NetFlow Configuration Guide at 2.

3209. Further examples include NetFlow Solutions Guide at § NetFlow Policy Routing.

2. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*expressed as a policy.*

3210. Cisco NetFlow discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the agent is expressed as a policy.” ’730 Pat. at Cl. 2. For example:

NetFlow can capture data over a long period of time, which enables you to track and anticipate network growth and plan upgrades. NetFlow service data can be used to optimize network planning, which includes peering, backbone upgrade planning, and routing policy planning. It also enables you to minimize the total cost of network operations while maximizing network performance, capacity, and reliability. NetFlow detects unwanted WAN traffic, validates bandwidth and quality of service (QoS) usage, and enables the analysis of new network applications. NetFlow offers valuable information that you can use to reduce the cost of operating the network.

NetFlow Configuration Guide at 2.

3211. Further examples include NetFlow Solutions Guide at § NetFlow Policy Routing.

**3. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3212. Cisco NetFlow discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. For example:

Network monitoring—NetFlow data enables extensive near real-time network monitoring capabilities. You can use NetFlow flow data analysis to display traffic patterns associated with individual routing devices and switches as well as on a networkwide basis (providing aggregate traffic or application-based views) to provide proactive problem detection, efficient troubleshooting, and rapid problem resolution.

NetFlow Solutions Guide at § Why Deploy NetFlow?

3213. Further examples include NetFlow Solutions Guide at § Capturing NetFlow Data, ISP Essentials at 44, NetFlow Configuration Guide at 2, Cyber Ops at 78.

- b. *constructing a topological representation of the computer network from the information.*

3214. Cisco NetFlow discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. For example:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Figure 21 shows a sample topology. To capture the flow of traffic going to site 2 of VPN 1 from any remote VPN 1 sites, you enable MPLS Egress NetFlow Accounting on link PE2-CE5 of provider edge router PE2. The flows are stored in a global flow cache maintained by the routing device. You can use the show ip cache flow command or other aggregation flow commands to view the egress flow data.

NetFlow Solutions Guide at § MPLS Egress NetFlow Accounting.

4. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3215. Cisco NetFlow discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. For example:

Neighbour authentication can be configured for the following router protocols: . . . Open Shortest Path First . . .

ISP Essentials at 60–61.

3216. Further, numerical models such as the Dijkstra self-stabilization algorithm were wellknown at the time of the invention. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm in view of a person of ordinary skill in the art.

5. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3217. Cisco NetFlow discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. For example:

Neighbour authentication can be configured for the following router protocols: . . . Open Shortest Path First . . .

ISP Essentials at 60–61.

3218. Further, numerical models such as the Dijkstra self-stabilization algorithm were wellknown at the time of the invention. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm in view of a person of ordinary skill in the art.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6. *Claim 7*

a. *A computer network, comprising:*

3219. Cisco NetFlow discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. For example:

NetFlow is a Cisco IOS application that provides statistics on packets flowing through the routing devices in the network. It is emerging as a primary network accounting and security technology.

NetFlow Configuration Guide at 1.

3220. Further examples include NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1).

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3221. Cisco NetFlow discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f.

c. *wherein the software agent has its own runtime environment*

3222. Cisco NetFlow discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

d. *is able to communicate with other software agents in the computer network*

3223. Cisco NetFlow discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c.

e. *is capable of perceiving its own state; and*

3224. Cisco NetFlow discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

f. *is able to clone itself;*

3225. Cisco NetFlow discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3226. Cisco NetFlow discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. For example:

User monitoring and profiling—NetFlow data enables you to gain detailed understanding of customer/user usage of network and application resources. This information may then be used to efficiently plan and allocate access, backbone, and application resources as well as to detect and resolve potential security and policy violations.

NetFlow Solutions Guide at § Why Deploy NetFlow?

3227. Further examples include NetFlow Solutions Guide at § NetFlow Infrastructure.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3228. Cisco NetFlow discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3229. Cisco NetFlow discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i.

j. *wherein the modeler determines appropriate policy based on the prediction;*

3230. Cisco NetFlow discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j.

k. *a network control mechanism to dynamically modify the assigned*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*goal of the software agent by replacing the assigned goal based on the optimal policy;*

3231. Cisco NetFlow discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3232. Cisco NetFlow discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l.

7. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3233. Cisco NetFlow discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g.

8. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

3234. Cisco NetFlow discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. For example:

NetFlow Data Export—Capture NetFlow accounting statistics for unicast ingress traffic or MPLS egress traffic on your networking device, and export your data to a collection device. Expired packet streams or flows are grouped together into "NetFlow Export" User Datagram Protocol (UDP) datagrams for export to a collection device. Using UDP datagrams, you provide simplicity and speed to your network compared to TCP networks where each packet is acknowledged. NetFlow allows you to aggregate

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

NetFlow data on the routing device before exporting to a collection device, resulting in lower bandwidth requirements for NetFlow data and reduced platform requirements for NetFlow data collection devices.

NetFlow Solutions Guide at § NetFlow Infrastructure.

9. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3235. Cisco NetFlow discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl.

12. For example:

NMS --network management system. A system responsible for managing at least part of a network. An NMS is generally a reasonably powerful and well-equipped computer, such as an engineering workstation. NMSs communicate with agents to help keep track of network statistics and resources.

NetFlow Configuration Guide at 218.

3236. Further examples include NetFlow Configuration Guide at 198, NetFlow Configuration Guide at 221, NetFlow Configuration Guide at 156.

10. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3237. Cisco NetFlow discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. For example:

NetFlow is a Cisco IOS application that provides statistics on packets flowing through the routing devices in the network. It is emerging as a primary network accounting and security technology.

NetFlow Configuration Guide at 1.

3238. Further examples include NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1).

11. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*environment controls an operation of the software agent based on predetermined criteria.*

3239. Cisco NetFlow discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. For example:

Enabling NetFlow on routers provides network administrators with access to “packet flow” information from their network. Exported NetFlow data can be used for a variety of purposes, including security monitoring, network management, capacity planning, customer billing, and Internet traffic flow analysis.

ISP Essentials at 44.

3240. Further examples include NetFlow Solutions Guide at § Why Deploy NetFlow?, NetFlow Solutions Guide at § Capturing NetFlow Data, NetFlow Configuration Guide at 2, Cyber Ops at 78.

12. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3241. Cisco NetFlow discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. For example:

The NetFlow MIB feature allows NetFlow statistics and other NetFlow data for the managed devices on your system to be retrieved by SNMP. You can specify retrieval of NetFlow information from a managed device (for example, a router) either by entering commands on that managed device or by entering SNMP commands from the NMS workstation to configure the router via the MIB. If the NetFlow information is configured from the NMS workstation, no access to the router is required and all configuration can be performed via SNMP. The NetFlow MIB request for information is sent from an NMS workstation via SNMP to the router and is retrieved from the router. This information can then be stored or viewed, thus allowing NetFlow information to be easily accessed and transported across a multi-vendor programming environment.

NetFlow Configuration Guide at 198.

3242. Further examples include NetFlow Configuration Guide at 221, NetFlow Configuration Guide at 218.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

13. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3243. Cisco NetFlow discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. For example:

The NetFlow MIB feature allows NetFlow statistics and other NetFlow data for the managed devices on your system to be retrieved by SNMP. You can specify retrieval of NetFlow information from a managed device (for example, a router) either by entering commands on that managed device or by entering SNMP commands from the NMS workstation to configure the router via the MIB. If the NetFlow information is configured from the NMS workstation, no access to the router is required and all configuration can be performed via SNMP. The NetFlow MIB request for information is sent from an NMS workstation via SNMP to the router and is retrieved from the router. This information can then be stored or viewed, thus allowing NetFlow information to be easily accessed and transported across a multi-vendor programming environment.

NetFlow Configuration Guide at 198.

3244. Further examples include NetFlow Configuration Guide at 221, NetFlow Configuration Guide at 218.

14. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3245. Cisco NetFlow discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. For example:

In this example, the first packet is being policy routed via route map test sequence 10. The subsequent packets of the same flow always take the same route map test sequence 10, not route map test sequence 20, because they all match or pass access list 1 check. Policy routing can be flow accelerated by bypassing the ACL check. The NetFlow cache entry size will be slightly extended (increased from 64 bytes to 96 bytes for every single flow) to reserve extra space per flow for state information for other features. Features such as setting policy routing extra proprietary fields, setting packet precedence, and setting the next-hop address will be stored in the state information per flow. The flow state information is not visible and will not appear using the show ip cache verbose flow command.

NetFlow Solutions Guide at § NPR Benefits.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3246. Further examples include NetFlow Configuration Guide at 160.

15. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3247. Cisco NetFlow discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. For example:

The Analyzer Display module incorporates a powerful and extensive graphical user interface (GUI) that enables you to: Configure network routing devices for NDE or TMS data export. Configure and control the operation of NFC workstations in collecting and storing NetFlow traffic data exported from NetFlow export-enabled devices in your network. Initiate and control the collection and storage of traffic matrix statistics (TMS) data exported from TMS export-enabled devices in your network. Retrieve stored NetFlow data from a designated NFC workstation and display the data in a selected format. Retrieve stored TMS data from an NFS-mounted storage volume in the network and display the data in a selected format.

NetFlow Solutions Guide at § Network Data Analyzer.

16. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3248. Cisco NetFlow discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6.

17. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3249. Cisco NetFlow discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. For example:

In this example, the first packet is being policy routed via route map test sequence 10. The subsequent packets of the same flow always take the same route map test sequence 10, not route map test sequence 20, because they

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

all match or pass access list 1 check. Policy routing can be flow accelerated by bypassing the ACL check. The NetFlow cache entry size will be slightly extended (increased from 64 bytes to 96 bytes for every single flow) to reserve extra space per flow for state information for other features. Features such as setting policy routing extra proprietary fields, setting packet precedence, and setting the next-hop address will be stored in the state information per flow. The flow state information is not visible and will not appear using the show ip cache verbose flow command.

NetFlow Solutions Guide at § NPR Benefits.

3250. Further examples include NetFlow Configuration Guide at 160. *See also* Claim Limitation 1.7, 1.8, 1.9, and 1.10.

18. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3251. Cisco NetFlow discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. For example:

User monitoring and profiling—NetFlow data enables you to gain detailed understanding of customer/user usage of network and application resources. This information may then be used to efficiently plan and allocate access, backbone, and application resources as well as to detect and resolve potential security and policy violations.

NetFlow Solutions Guide at § Why Deploy NetFlow?

3252. Further examples include NetFlow Solutions Guide at § Capturing NetFlow Data, ISP Essentials at 44, NetFlow Configuration Guide at 2, Cyber Ops at 78.

19. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3253. Cisco NetFlow discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. For example:

“NetFlow is a Cisco IOS application that provides statistics on packets flowing through the routing devices in the network. It is emerging as a primary network accounting and security technology.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

NetFlow Configuration Guide at 1.

3254. Further examples include NetFlow Solutions Guide at § NetFlow Infrastructure (Fig. 1).

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3255. Cisco NetFlow discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b.

- c. *is able to communicate with other software agents in the computer network*

3256. Cisco NetFlow discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c.

- d. *is capable of perceiving its own state; and*

3257. Cisco NetFlow discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d.

- e. *is able to clone itself;*

3258. Cisco NetFlow discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e.

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3259. Cisco NetFlow discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3260. Cisco NetFlow discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3261. Cisco NetFlow discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3262. Cisco NetFlow discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3263. Cisco NetFlow discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3264. Cisco NetFlow discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k.

20. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3265. Cisco NetFlow discloses this element of claim 31, “[t]he machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy” at least for all the reasons explained above regarding claim 2, element a.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

21. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3266. Cisco NetFlow discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a.

- b. *constructing a topological representation of the computer network from the information.*

3267. Cisco NetFlow discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b.

22. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3268. Cisco NetFlow discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3269. Cisco NetFlow discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k.

23. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*determining uses a numerical method.*

3270. Cisco NetFlow discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4.

24. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3271. Cisco NetFlow discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6.

3272. I expect to testify that Cisco NetFlow anticipates and/or renders obvious each Asserted Claim of the ’730 Patent. A claim chart illustrating that Cisco NetFlow discloses and/or renders obvious each and every limitation of those claims is included as Exhibit A-25.

**XI. PRIOR ART COMBINATIONS RENDERING THE ’730 PATENT OBVIOUS**

**A. NetRanger in Combination with Goldman**

3273. Under NetFuel’s interpretation of the claims, Cisco NetRanger in combination with Goldman renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, software network management. NetRanger discusses automatically configuring security policies, and Goldman discloses testing policies prior to deployment to avoid incorrect configuration, inconsistent policies, and other device failures. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco NetRanger and Goldman. It would have been further obvious to combine the teachings of NetRanger and Goldman because the combination is merely the use of a known technique to improve a similar device, method, or product in the same way. It is my opinion that, under NetFuel’s interpretation of the claims, NetRanger in combination

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

with Goldman discloses and/or renders obvious all elements of the '730 Patent. *See* Ex. A-3, Ex. A-8.

**B. Cisco Catalyst 5000 Switch Family in combination with NetRanger and/or Goldman**

3274. It would have been obvious to combine the Cisco Catalyst 5000 Switch Family with NetRanger and Goldman, because the combination is merely the use of a known technique to improve a similar device, method, or product in the same way. NetRanger discloses automatically configuring security policies in response to detected events by reconfiguring the Access Control Lists (ACLs) of a Cisco switch, and the Catalyst 5000 Switch family contains ACLs for security and QoS matching. Goldman discloses testing policies prior to deployment to avoid incorrect configuration, inconsistent policies, and other device failures. Cisco Catalyst 5000 Switch Family in combination with NetRanger and/or Goldman discloses and/or renders obvious all elements of the '730 Patent. Ex. A-9. Cisco Catalyst 5000 Switch Family, NetRanger, and Goldman are all in the field of software network management.

**C. Goldman in combination with Kasteleijn and/or Natarajan and/or RFC 2328**

**1. *Motivation to Combine Goldman with Kasteleijn***

3275. A person of ordinary skill in the art would understand that an agent capable of communicating with other agents in improving the efficiency of agents in a distributed network. Further, It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

**2. *Motivation to Combine Goldman with Natarajan***

3276. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Goldman's agents to request further policy from another agent when it cannot resolve a fault or satisfy a performance target.

**3. *Motivation to Combine Goldman with RFC 2328***

3277. Goldman discloses target configuration agents testing and deploying policies on network targets. It would have been obvious at the time of invention to use path information or

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

other information from the target agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSFP messages to update the OSPF routing table. Further, it would have been obvious in combination with RFC 2328 to trigger events based off OSPF state

4. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3278. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3279. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

3280. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1.

d. *is capable of perceiving its own state*

3281. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1.

e. *and is able to clone itself,*

3282. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent; and*

3283. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1.

g. *monitoring the computer network;*

3284. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3285. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

3286. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3287. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3288. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3289. Goldman discloses this claim element. *See* Goldman Anticipation Claim 1. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Natarajan discloses this claim element. '730 Pat. at Cl. 1; *see* Goldman Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-7, claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

5. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3290. Goldman discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the Goldman discloses this claim element. *See* Goldman Anticipation Claim 2.

6. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3291. Goldman discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* Goldman Anticipation Claim 1.

3292. To the extent this limitation is not disclosed by Goldman, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2). Goldman discloses target configuration agents testing and deploying policies on network targets. It would have been obvious at the time of invention to use path information or other information from the target agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSFP messages to update the OSPF routing table.

- b. *constructing a topological representation of the computer network from the information.*

3293. Goldman discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Goldman Anticipation Claim 3.

3294. To the extent this limitation is not disclosed by Goldman, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2). Goldman discloses target configuration agents testing and deploying policies on network targets. It would have been obvious at the time of invention to use path information or other information from the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

target agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSFP messages to update the OSPF routing table.

7. **Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3295. Goldman discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Goldman Anticipation Claim 4.

3296. To the extent this limitation is not disclosed by Goldman, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

3297. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Pandya’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

8. **Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Dijkstra Self Stabilization Algorithm.*

3298. Goldman discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Goldman Anticipation Claim 6.

3299. To the extent Goldman does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Goldman in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* Goldman Anticipation Claims 1, 6; Goldman Combination Claim 1; Ex. A-3, claim 6.

3300. Goldman discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Goldman Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3301. To the extent this limitation is not expressly disclosed by Goldman, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. Goldman discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3302. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Goldman’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to***



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3303. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Goldman to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Goldman's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

9. ***Claim 7***

a. *A computer network, comprising:*

3304. Goldman discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* Goldman Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3305. Goldman discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See* Goldman Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3306. Goldman discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See* Goldman Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

3307. Goldman discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Goldman Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3308. Goldman discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Goldman Anticipation Claim 7.

f. *is able to clone itself;*

3309. Goldman discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Goldman Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3310. Goldman discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Goldman Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3311. Goldman discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Goldman Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3312. Goldman discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Goldman Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

3313. Goldman discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Goldman Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3314. Goldman discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Goldman Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3315. Goldman discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Goldman Anticipation Claim 7.

10. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3316. Goldman discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Goldman Anticipation Claim 10.

11. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

3317. Goldman discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Goldman Anticipation Claim 11. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 11; *see* Goldman Anticipation Claim 11; Ex. A-2, claim 11; Ex. A-3, claim 11.

12. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3318. Goldman discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Goldman Anticipation Claim 12.

13. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3319. Goldman discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Goldman Anticipation Claim 13. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 13; *see* Goldman Anticipation Claim 13; Ex. A-2, claim 13; Ex. A-3, claim 13.

14. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3320. Goldman discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Goldman Anticipation Claim 16.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

15. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

3321. Goldman discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Goldman Anticipation Claim 17.

3322. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 17; *see* Goldman Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-3, claim 17.

16. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3323. Goldman discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Goldman Anticipation Claim 18. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 18; *see* Goldman Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-3, claim 18.

17. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3324. Goldman discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Goldman Anticipation Claim 19. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 19; *see* Goldman Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-3, claim 19.

18. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*database to store the policy for the agent.*

3325. Goldman discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Goldman Anticipation Claim 21.

19. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3326. Goldman discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Goldman Anticipation Claim 22.

20. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3327. Goldman discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Goldman Anticipation Claim 24.

21. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3328. Goldman discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Goldman Anticipation Claim 26.

22. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3329. Goldman discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Goldman Anticipation Claim

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

29. To the extent Goldman does not by itself anticipate this claim element, Goldman in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 29; *see* Goldman Anticipation Claim 29; Ex. A-2, claim 29; Ex. A-3, claim 29.

23. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3330. Goldman discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Goldman Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3331. Goldman discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Goldman Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

3332. Goldman discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Goldman Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

3333. Goldman discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Goldman Anticipation Claim 30.

- e. *is able to clone itself;*

3334. Goldman discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Goldman Anticipation Claim 30.

- f. *and comprises an autonomous agent operable to request further*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy when it lacks an ability to perform the predefined task*

3335. Goldman discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l. *See* Goldman Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3336. Goldman discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Goldman Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3337. Goldman discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Goldman Anticipation Claim 30.

i. *including predicting a failure of a network component based on a predictive algorithm;*

3338. Goldman discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Goldman Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3339. Goldman discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Goldman Anticipation Claim 30.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

3340. Goldman discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Goldman Anticipation Claim 30.

24. ***Claim 31***

3341. Goldman discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Goldman Anticipation Claim 31.

25. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3342. Goldman discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Goldman Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3343. Goldman discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Goldman Anticipation Claim 32.

26. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3344. Goldman discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Goldman Anticipation Claim 33.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3345. Goldman discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Goldman Anticipation Claim 33.

27. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3346. Goldman discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Goldman Anticipation Claim 34.

28. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3347. Goldman discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Goldman Anticipation Claim 36.

**D. Pandya in combination with Kasteleijn and/or Goldman and/or Douik and/or RFC 2328/Afek**

1. ***Motivation to Combine Pandya with Kasteleijn***

3348. The motivation for this modification would be to permit agents to propagate throughout the network as taught by Kasteleijn.

2. ***Motivation to Combine Pandya with Goldman***

3349. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to the agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, failure to meet QoS requirements, insufficient bandwidth, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman.

3. ***Motivation to Combine Pandya with Douik***

3350. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (by monitoring the network for events that may indicate failure), where modeling includes predicting failure of a network component (such as congestion or other device failure via a rules in a diagnosis or correlation agent), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* preventative actions determined by the agents), as taught by Douik.

4. ***Motivation to Combine Pandya with RFC 2328 and/or Afek***

3351. Pandya in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Pandya discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Pandya and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Pandya because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Pandya are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Pandya to include these elements, rendering them obvious.

5. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3352. Pandya discloses this claim element. *See Pandya Anticipation Claim 1.*

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3353. Pandya discloses this claim element. *See Pandya Anticipation Claim 1.*

c. *is able to communicate with other software agents in the computer network;*

3354. Pandya discloses this claim element. *See Pandya Anticipation Claim 1.*

d. *is capable of perceiving its own state*

3355. Pandya discloses this claim element. *See Pandya Anticipation Claim 1.* To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see Pandya Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1.*

e. *and is able to clone itself,*

3356. Pandya discloses this claim element. *See Pandya Anticipation Claim 1.* To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see Pandya Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1.*

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3357. Pandya discloses this claim element. *See Pandya Anticipation Claim 1.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

g. *monitoring the computer network;*

3358. Pandya discloses this claim element. *See* Pandya Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3359. Pandya discloses this claim element. *See* Pandya Anticipation Claim 1. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Goldman and/or Douik discloses this claim element. '730 Pat. at Cl. 1; *see* Pandya Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

3360. Pandya discloses this claim element. *See* Pandya Anticipation Claim 1. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Goldman and/or Douik discloses this claim element. '730 Pat. at Cl. 1; *see* Pandya Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-4, claim 1; Ex. A-5, claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3361. Pandya discloses this claim element. *See* Pandya Anticipation Claim 1. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Goldman and/or Douik discloses this claim element. '730 Pat. at Cl. 1; *see* Pandya Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-4, claim 1; Ex. A-5, claim 1.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3362. Pandya discloses this claim element. *See* Pandya Anticipation Claim 1. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Goldman and/or Douik discloses this claim element. '730 Pat. at Cl. 1; *see* Pandya Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-4, claim 1; Ex. A-5, claim 1.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3363. Pandya discloses this claim element. *See* Pandya Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**6. Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3364. Pandya discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the aPandya discloses this claim element. *See Pandya Anticipation Claim 2.*

**7. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3365. Pandya discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See Pandya Anticipation Claim 1.*

- b. *constructing a topological representation of the computer network from the information.*

3366. Pandya discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See Pandya Anticipation Claim 3.*

3367. To the extent this limitation is not disclosed by Pandya, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2). Pandya discloses mobile agents collecting information for reporting back to a central manager. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. As each agent collects information about the network, it may collect information as to the endpoints and states of network links. This information may be used to substitute or supplement link state OSFP messages to update the OSPF routing table.

**8. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3368. Pandya discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See Pandya Anticipation Claim 4.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3369. To the extent this limitation is not disclosed by Pandya, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

3370. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Pandya's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

9. **Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3371. Pandya discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Pandya Anticipation Claim 6.

3372. To the extent Pandya does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Pandya in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* Pandya Anticipation Claims 1, 6; Pandya Combination Claim 1; Ex. A-4, claim 6.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3373. Pandya discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Pandya Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3374. To the extent this limitation is not expressly disclosed by Pandya, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '730 Patent. Pandya discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3375. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Pandya's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3376. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Pandya



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Pandya's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

10. ***Claim 7***

a. *A computer network, comprising:*

3377. Pandya discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See Pandya Anticipation Claim 7.*

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3378. Pandya discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See Pandya Anticipation Claim 7.*

c. *wherein the software agent has its own runtime environment*

3379. Pandya discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See Pandya Anticipation Claim 7.*

d. *is able to communicate with other software agents in the computer network*

3380. Pandya discloses this element of claim 7, "is able to communicate with other software agents in the computer network," at least for all the reasons explained above regarding claim 1, element c. *See Pandya Anticipation Claim 7.*

e. *is capable of perceiving its own state; and*

3381. Pandya discloses this element of claim 7, "is capable of perceiving its own state," at least for all the reasons explained above regarding claim 1, element d. *See Pandya Anticipation Claim 7.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

f. *is able to clone itself;*

3382. Pandya discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See Pandya Anticipation Claim 7.*

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3383. Pandya discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See Pandya Anticipation Claim 7.*

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3384. Pandya discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See Pandya Anticipation Claim 7.*

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3385. Pandya discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See Pandya Anticipation Claim 7.*

j. *wherein the modeler determines appropriate policy based on the prediction;*

3386. Pandya discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See Pandya Anticipation Claim 7.*

k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3387. Pandya discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Pandya Anticipation Claim 7.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3388. Pandya discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Pandya Anticipation Claim 7.

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3389. Pandya discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Pandya Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

3390. Pandya discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Pandya Anticipation Claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3391. Pandya discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Pandya Anticipation Claim 12.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3392. Pandya discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Pandya Anticipation Claim 13.

3393. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 12; *see* Pandya Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-4, claim 12.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3394. Pandya discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Pandya Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

3395. Pandya discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Pandya Anticipation Claim 17. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 17; *see* Pandya Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-4, claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3396. Pandya discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

See Pandya Anticipation Claim 18. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 18; see Pandya Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-4, claim 18.

18. **Claim 19**

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3397. Pandya discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. See Pandya Anticipation Claim 19. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 19; see Pandya Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-4, claim 19.

19. **Claim 21**

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3398. Pandya discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. See Pandya Anticipation Claim 21. To the extent Pandya does not by itself anticipate this claim element, Pandya in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 21; see Pandya Anticipation Claim 21; Ex. A-2, claim 21; Ex. A-4, claim 21.

20. **Claim 22**

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3399. Pandya discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. See Pandya Anticipation Claim 22.

21. **Claim 24**

- a. *The computer network of claim 7, wherein the modeler comprises*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*a Dijkstra Self Stabilization Algorithm.*

3400. Pandya discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Pandya Anticipation Claim 24.

22. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3401. Pandya discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Pandya Anticipation Claim 26.

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3402. Pandya discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Pandya Anticipation Claim 29.

24. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3403. Pandya discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Pandya Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3404. Pandya discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Pandya Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

3405. Pandya discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Pandya Anticipation Claim 30.

d. *is capable of perceiving its own state; and*

3406. Pandya discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Pandya Anticipation Claim 30.

e. *is able to clone itself;*

3407. Pandya discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Pandya Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3408. Pandya discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* Pandya Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3409. Pandya discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See* Pandya Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3410. Pandya discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Pandya Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3411. Pandya discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See Pandya Anticipation Claim 30.*

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3412. Pandya discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See Pandya Anticipation Claim 30.*

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3413. Pandya discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See Pandya Anticipation Claim 30.*

25. ***Claim 31***

3414. Pandya discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See Pandya Anticipation Claim 31.*

26. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3415. Pandya discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See Pandya Anticipation Claim 32.*

- b. *constructing a topological representation of the computer network*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

3416. Pandya discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Pandya Anticipation Claim 32.

**27. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3417. Pandya discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Pandya Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3418. Pandya discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Pandya Anticipation Claim 33.

**28. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3419. Pandya discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Pandya Anticipation Claim 34.

**29. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Algorithm.*

3420. Pandya discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Pandya Anticipation Claim 36.

**E. Douik in combination with Kasteleijn and/or Turek and/or Afek and/or Goldman**

**1. *Motivation to Combine Douik with Kasteleijn***

3421. Douik discloses autonomous agents, which would need goals and a runtime environment like the agents in Kasteleijn. Further, autonomous agents that cooperate to monitor different components of a network would clone themselves if the situation required it (such as if a new network component were installed and additional agents were needed to monitor it). Furthermore, this limitation is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4. It would have been obvious to permit software agents to clone themselves as the agents in Kasteleijn are cloned in order to, for example, permit an increased flexibility in the size of the network that is monitored.

**2. *Motivation to Combine Douik with Turek***

3422. Douik discloses using agents to gather information about the network, and then using that information to construct a representation of the connectivity of the network. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table. Douik also discloses using agents to gather information about the network, and then using that information to construct a representation of the connectivity of the network. It would have been obvious at the time of invention to use the OSPF algorithm to model the network. Further, it would have been obvious at the time of invention to one of ordinary skill in the art to enable Douik’s agents to request further policy from another entity when it cannot resolve a fault or satisfy a performance target.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3. ***Motivation to Combine Douik with Afek***

3423. Modeling using numerical models such as the Dijkstra self-stabilization algorithm were well known at the time of invention. It would have been obvious, and merely a design choice, to create a model that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek.

4. ***Motivation to Combine Douik with Goldman***

3424. A person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer “modified” but considered “new.”

5. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3425. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3426. Douik discloses this claim element. *See* Douik Anticipation Claim 1. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Douik Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-5, claim 1.

c. *is able to communicate with other software agents in the computer network;*

3427. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

d. *is capable of perceiving its own state*

3428. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

e. *and is able to clone itself,*

3429. Douik discloses this claim element. *See* Douik Anticipation Claim 1. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Douik Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-5, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent; and*

3430. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

g. *monitoring the computer network;*

3431. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3432. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

3433. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3434. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3435. Douik discloses this claim element. *See* Douik Anticipation Claim 1.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3436. Douik discloses this claim element. *See* Douik Anticipation Claim 1. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Turek discloses this claim element. '730 Pat. at Cl. 1; *see* Douik Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-5 claim 1.

**6. Claim 2**

a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3437. Douik discloses claim 2, "[t]he method of claim 2, wherein the assigned goal of the aDouik discloses this claim element. *See* Douik Anticipation Claim 2.

**7. Claim 3**

a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the predefined task; and*

3438. Douik discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* Douik Anticipation Claim 1.

b. *constructing a topological representation of the computer network from the information.*

3439. Douik discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Douik Anticipation Claim 3.

3440. To the extent this limitation is not disclosed by Douik, it would have been obvious to combine Douik with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). Douik discloses using agents to gather information about the network, and then using that information to construct a representation of the connectivity of the network. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

8. ***Claim 4***

a. *The method of claim 1, wherein the modeling uses a numerical method.*

3441. Douik discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Douik Anticipation Claim 4.

3442. To the extent this limitation is not disclosed by Douik, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

3443. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Pandya’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. **The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

9. **Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3444. Douik discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Douik Anticipation Claim 6.

3445. To the extent Douik does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Douik in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* Douik Anticipation Claims 1, 6; Douik Combination Claim 1; Ex. A-5, claim 6.

3446. Douik discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Douik Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3447. To the extent this limitation is not expressly disclosed by Douik, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '730 Patent. Douik discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3448. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Douik's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3449. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Douik to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Douik's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

10. ***Claim 7***

a. *A computer network, comprising:*

3450. Douik discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* Douik Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*hardware*

3451. Douik discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Douik Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3452. Douik discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Douik Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

3453. Douik discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Douik Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3454. Douik discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Douik Anticipation Claim 7.

f. *is able to clone itself;*

3455. Douik discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Douik Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3456. Douik discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Douik Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*thereby to determine an optimal policy for the computer network*

3457. Douik discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Douik Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3458. Douik discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Douik Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

3459. Douik discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Douik Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3460. Douik discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Douik Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3461. Douik discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Douik Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3462. Douik discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Douik Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

3463. Douik discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Douik Anticipation Claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3464. Douik discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Douik Anticipation Claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3465. Douik discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Douik Anticipation Claim 13. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 12; *see* Douik Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-5, claim 12.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3466. Douik discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Douik Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

3467. Douik discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Douik Anticipation Claim 17. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 17; *see* Douik Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-5, claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3468. Douik discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Douik Anticipation Claim 18. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 18; *see* Douik Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-5, claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3469. Douik discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Douik Anticipation Claim 19.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

19. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3470. Douik discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Douik Anticipation Claim 21.

20. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3471. Douik discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Douik Anticipation Claim 22.

21. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3472. Douik discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Douik Anticipation Claim 24.

22. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3473. Douik discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Douik Anticipation Claim 26. To the extent Douik does not by itself anticipate this claim element, Douik in combination with Goldman discloses this claim element. ’730 Pat. at Cl. 26; *see* Douik Anticipation Claim 26; Ex. A-3, claim 26; Ex. A-5, claim 26.

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

3474. Douik discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Douik Anticipation Claim 29.

24. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3475. Douik discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Douik Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3476. Douik discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Douik Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

3477. Douik discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Douik Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

3478. Douik discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Douik Anticipation Claim 30.

- e. *is able to clone itself;*

3479. Douik discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Douik Anticipation Claim 30.

- f. *and comprises an autonomous agent operable to request further*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy when it lacks an ability to perform the predefined task*

3480. Douik discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l. *See* Douik Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3481. Douik discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Douik Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3482. Douik discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Douik Anticipation Claim 30.

i. *including predicting a failure of a network component based on a predictive algorithm;*

3483. Douik discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Douik Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3484. Douik discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Douik Anticipation Claim 30.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

3485. Douik discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Douik Anticipation Claim 30.

25. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3486. Douik discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Douik Anticipation Claim 31.

26. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3487. Douik discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Douik Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3488. Douik discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Douik Anticipation Claim 32.

27. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3489. Douik discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Douik Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3490. Douik discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Douik Anticipation Claim 33.

28. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3491. Douik discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Douik Anticipation Claim 34.

29. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3492. Douik discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Douik Anticipation Claim 36.

**F. Natarajan in combination with Kasteleijn and/or Goldman and/or Turek and/or Yoshihara and/or RFC 2328**

1. ***Motivation to Combine Natarajan with Kasteleijn***

3493. Natarajan discloses autonomous agents, which would need goals and a runtime environment like the agents in Kasteleijn. Further, autonomous agents that cooperate to monitor different components of a network would clone themselves if the situation required it (such as if a new network component were installed and additional agents were needed to monitor it).



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Furthermore, this limitation is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4. It would have been obvious to permit software agents to clone themselves as the agents in Kasteleijn are cloned in order to, for example, permit an increased flexibility in the size of the network that is monitored.

**2. *Motivation to Combine Natarajan with Goldman***

3494. Natarajan discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not violate the predetermined values), where modeling includes predicting failure of a network component (e.g. incorrect configuration, violation of predetermined values, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. modified or new policies), as taught by Goldman.

**3. *Motivation to Combine Natarajan with Turek***

3495. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Natarajan's network elements or monitor system to request further policy from the policy engine or the ADMIN system when it cannot resolve a fault or satisfy a performance target.

**4. *Motivation to Combine Natarajan with Yoshihara***

3496. It would have been obvious at the time of invention to a skilled artisan to employ Natarajan's network elements or monitor system to request further policy from the policy engine or the ADMIN system when it cannot solve an issue or satisfy a performance target.

**5. *Motivation to Combine Natarajan with RFC 2328***

3497. Natarajan discloses monitoring different characteristics of the network. It would have been obvious to one of ordinary skill in the art to monitor network characteristics also used by OSPF, such as routing table changes, link state messages, or changes in link status, and triggering events off those characteristics. This satisfies the limitation under NetFuel's apparent

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

interpretation of the claims. It would have also been obvious to utilize information gathered by the monitoring agents to supplement or replace link state messages as used by OSPF to build its routing table (i.e., network topology).

6. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3498. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.*

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3499. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.*

c. *is able to communicate with other software agents in the computer network;*

3500. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.*

d. *is capable of perceiving its own state*

3501. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.*

e. *and is able to clone itself,*

3502. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.*

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3503. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.*

g. *monitoring the computer network;*

3504. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.*

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3505. Natarajan discloses this claim element. *See Natarajan Anticipation Claim 1.* To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Goldman discloses this claim element. '730 Pat. at Cl. 1; *see Natarajan Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-7, claim 1.*

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***prediction algorithm*

3506. Natarajan discloses this claim element. *See* Natarajan Anticipation Claim 1. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Goldman discloses this claim element. '730 Pat. at Cl. 1; *see* Natarajan Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-7, claim 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3507. Natarajan discloses this claim element. *See* Natarajan Anticipation Claim 1. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Goldman discloses this claim element. '730 Pat. at Cl. 1; *see* Natarajan Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-7, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3508. Natarajan discloses this claim element. *See* Natarajan Anticipation Claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3509. Natarajan discloses this claim element. *See* Natarajan Anticipation Claim 1. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Turek and/or Yoshihara discloses this claim element. '730 Pat. at Cl. 1; *see* Natarajan Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-7, claim 1; Ex. A-21, claim 1.

7. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3510. Natarajan discloses claim 2, "[t]he method of claim 2, wherein the assigned goal of the aNatarajan discloses this claim element. *See* Natarajan Anticipation Claim 2. To the extent this limitation is not disclosed in Natarajan it would have been obvious in view of RFC 2328 (OSPF). Natarajan discloses monitoring different characteristics of the network. It would have been obvious to one of ordinary skill in the art to monitor network characteristics also used by OSFP, such as routing table changes, link state messages, or changes in link status, and triggering

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

events off those characteristics. This satisfies the limitation under NetFuel's apparent interpretation of the claims. It would have also been obvious to utilize information gathered by the monitoring agents to supplement or replace link state messages as used by OSPF to build its routing table (i.e., network topology).

8. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3511. Natarajan discloses this element of claim 3, "[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. *See* Natarajan Anticipation Claim 1. To the extent this limitation is not disclosed in Natarajan it would have been obvious in view of RFC 2328 (OSPF). Natarajan discloses monitoring different characteristics of the network. It would have been obvious to one of ordinary skill in the art to monitor network characteristics also used by OSPF, such as routing table changes, link state messages, or changes in link status, and triggering events off those characteristics. This satisfies the limitation under NetFuel's apparent interpretation of the claims. It would have also been obvious to utilize information gathered by the monitoring agents to supplement or replace link state messages as used by OSPF to build its routing table (i.e., network topology).

- b. *constructing a topological representation of the computer network from the information.*

3512. Natarajan discloses this element of claim 3, "constructing a topological representation of the computer network from the information." '730 Pat. at Cl. 3. *See* Natarajan Anticipation Claim 3.

9. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3513. Natarajan discloses claim 4, "[t]he method of claim 1, wherein the modeling uses a numerical method." '730 Pat. at Cl. 4. *See* Natarajan Anticipation Claim 4.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**10. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3514. Natarajan discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Natarajan Anticipation Claim 6. To the extent this limitation is not disclosed in Natarajan it would have been obvious in view of RFC 2328 (OSPF). Natarajan discloses monitoring different characteristics of the network. It would have been obvious to one of ordinary skill in the art to monitor network characteristics also used by OSPF, such as routing table changes, link state messages, or changes in link status, and triggering events off those characteristics. This satisfies the limitation under NetFuel’s apparent interpretation of the claims. It would have also been obvious to utilize information gathered by the monitoring agents to supplement or replace link state messages as used by OSPF to build its routing table (i.e., network topology).

11. ***Claim 7***

- a. *A computer network, comprising:*

3515. Natarajan discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* Natarajan Anticipation Claim 7.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3516. Natarajan discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Natarajan Anticipation Claim 7.

- c. *wherein the software agent has its own runtime environment*

3517. Natarajan discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Natarajan Anticipation Claim 7.

- d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

3518. Natarajan discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Natarajan Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3519. Natarajan discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Natarajan Anticipation Claim 7.

f. *is able to clone itself;*

3520. Natarajan discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Natarajan Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3521. Natarajan discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Natarajan Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3522. Natarajan discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Natarajan Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3523. Natarajan discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Natarajan Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

3524. Natarajan discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Natarajan Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3525. Natarajan discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Natarajan Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3526. Natarajan discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Natarajan Anticipation Claim 7.

12. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3527. Natarajan discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Natarajan Anticipation Claim 10.

13. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

3528. Natarajan discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Natarajan Anticipation Claim 11. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 11; *see* Natarajan Anticipation Claim 11; Ex. A-2, claim 11; Ex. A-7, claim 11.

14. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3529. Natarajan discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Natarajan Anticipation Claim 12.

15. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3530. Natarajan discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Natarajan Anticipation Claim 13. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 13; *see* Natarajan Anticipation Claim 13; Ex. A-2, claim 13; Ex. A-7, claim 13.

16. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3531. Natarajan discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Natarajan Anticipation Claim 16.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**17. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

3532. Natarajan discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Natarajan Anticipation Claim 17. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 17; *see* Natarajan Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-7, claim 17.

18. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3533. Natarajan discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Natarajan Anticipation Claim 18. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 18; *see* Natarajan Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-7, claim 18.

19. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3534. Natarajan discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Natarajan Anticipation Claim 19. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 19; *see* Natarajan Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-7, claim 19.

20. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*database to store the policy for the agent.*

3535. Natarajan discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Natarajan Anticipation Claim 21.

21. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3536. Natarajan discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Natarajan Anticipation Claim 22. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Yoshihara discloses this claim element. ’730 Pat. at Cl. 22; *see* Natarajan Anticipation Claim 22; Ex. A-7, claim 22; Ex. A-21, claim 22.

22. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3537. Natarajan discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Natarajan Anticipation Claim 24.

23. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3538. Natarajan discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Natarajan Anticipation Claim 26.

24. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

3539. Natarajan discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Natarajan Anticipation Claim 29. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 29; *see* Natarajan Anticipation Claim 29; Ex. A-2, claim 29; Ex. A-7, claim 29.

25. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3540. Natarajan discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Natarajan Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3541. Natarajan discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Natarajan Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

3542. Natarajan discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Natarajan Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

3543. Natarajan discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Natarajan Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is able to clone itself;*

3544. Natarajan discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Natarajan Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3545. Natarajan discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* Natarajan Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3546. Natarajan discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See* Natarajan Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3547. Natarajan discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Natarajan Anticipation Claim 30.

i. *including predicting a failure of a network component based on a predictive algorithm;*

3548. Natarajan discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Natarajan Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

3549. Natarajan discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Natarajan Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3550. Natarajan discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Natarajan Anticipation Claim 30.

26. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3551. Natarajan discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Natarajan Anticipation Claim 31.

27. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3552. Natarajan discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Natarajan Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3553. Natarajan discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Natarajan Anticipation Claim 32.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

28. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3554. Natarajan discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Natarajan Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3555. Natarajan discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Natarajan Anticipation Claim 33.

29. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3556. Natarajan discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Natarajan Anticipation Claim 34.

30. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3557. Natarajan discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Natarajan Anticipation Claim 36.

**G. Bonnell in combination with Kasteleijn and/or Goldman and/or Hellerstein**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****and/or Turk and/or RFC 2328/Afek****1. *Motivation to Combine Bonnell with Kasteleijn***

3558. It would have been obvious to permit software agents to communicate amongst each others, as the MCs of Kasteleijn do, in order to permit code or functionality downloads from one agent to another, or generally exchange messages (such as informing other agents that a console is requesting a certain type of information). Also, it would have been obvious to have the agents clone themselves, for example, when the size of the network grows. Further, this limitation is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4. It would have been obvious to permit software agents to clone themselves as the agents in Kasteleijn are cloned in order to, for example, permit an increased flexibility in the size of the network that is monitored. Further, It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

**2. *Motivation to Combine Bonnell with Goldman***

3559. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to Bonnell's agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* additional code or agents), as taught by Goldman. A person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer "modified" but considered "new."

**3. *Motivation to Combine Bonnell with Hellerstein***

3560. Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine Bonnell and Hellerstein to implement these models.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****4. Motivation to Combine Bonnell with Turek**

3561. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Bonnell's agents to request further policy from another agent, collector agent, or network manager when it cannot resolve a fault or satisfy a performance target.

**5. Motivation to Combine Motivation to Combine Bonnell with RFC 2328 and/or Afek**

3562. Bonnell in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Bonnell discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Bonnell and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Bonnell because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Bonnell are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Bonnell to include these elements, rendering them obvious.

**6. Claim 1****a. A method of managing a computer network, comprising:**

3563. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1.

**b. assigning a goal to a software [agent], wherein the software agent**



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*has its own runtime environment*

3564. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

3565. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-15, claim 1.

d. *is capable of perceiving its own state*

3566. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1.

e. *and is able to clone itself,*

3567. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-15, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3568. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1.

g. *monitoring the computer network;*

3569. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3570. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-15, claim 1; Ex. A-20, claim 1.

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***prediction algorithm*

3571. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-15, claim 1; Ex. A-20, claim 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3572. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-15, claim 1; Ex. A-20, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3573. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-15, claim 1; Ex. A-20, claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3574. Bonnell discloses this claim element. *See* Bonnell Anticipation Claim 1. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Turek discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-15, claim 1.

7. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3575. Bonnell discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the aBonnell discloses this claim element. *See* Bonnell Anticipation Claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

8. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3576. Bonnell discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* Bonnell Anticipation Claim 1.

- b. *constructing a topological representation of the computer network from the information.*

3577. Bonnell discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Bonnell Anticipation Claim 3.

3578. To the extent this limitation is not disclosed by Bonnell, it would have been obvious to combine Bonnell with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). Bonnell discloses agents monitoring the network. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. Each agent may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

9. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3579. Bonnell discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Bonnell Anticipation Claim 4.

3580. To the extent this limitation is not disclosed by Bonnell, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3581. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Pandya's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

10. **Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3582. Bonnell discloses claim 6, "[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm." '730 Pat. at Cl. 6. *See* Bonnell Anticipation Claim 6.

3583. To the extent Bonnell does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Bonnell in combination with RFC 2328 and/or Afek discloses this claim element. '730 Pat. at Cl. 1; *see* Bonnell Anticipation Claims 1, 6; Bonnell Combination Claim 1; Ex. A-15, claim 6.

3584. Bonnell discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Bonnell Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3585. To the extent this limitation is not expressly disclosed by Bonnell, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '730 Patent. Bonnell discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3586. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Bonnell's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3587. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Bonnell to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Bonnell's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 7***

a. *A computer network, comprising:*

3588. Bonnell discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* Bonnell Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3589. Bonnell discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Bonnell Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3590. Bonnell discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Bonnell Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

3591. Bonnell discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Bonnell Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3592. Bonnell discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Bonnell Anticipation Claim 7.

f. *is able to clone itself;*

3593. Bonnell discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Bonnell Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

3594. Bonnell discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Bonnell Anticipation Claim 7.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3595. Bonnell discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Bonnell Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3596. Bonnell discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Bonnell Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

3597. Bonnell discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Bonnell Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3598. Bonnell discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Bonnell Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

3599. Bonnell discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Bonnell Anticipation Claim 7.

12. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3600. Bonnell discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Bonnell Anticipation Claim 10.

13. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

3601. Bonnell discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Bonnell Anticipation Claim 11. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 11; *see* Bonnell Anticipation Claim 11; Ex. A-2, claim 11; Ex. A-15, claim 11.

14. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3602. Bonnell discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Bonnell Anticipation Claim 12. To the extent Bonnell does not by itself anticipate this claim



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

element, Bonnell in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 12; *see* Bonnell Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-15, claim 12.

15. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3603. Bonnell discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Bonnell Anticipation Claim 13.

16. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3604. Bonnell discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Bonnell Anticipation Claim 16.

17. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

3605. Bonnell discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Bonnell Anticipation Claim 17.

18. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3606. Bonnell discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Bonnell Anticipation Claim 18. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 18; *see* Bonnell Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-15, claim 18.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

19. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3607. Bonnell discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Bonnell Anticipation Claim 19. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 19; *see* Bonnell Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-15, claim 19.

20. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3608. Bonnell discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Bonnell Anticipation Claim 21.

21. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3609. Bonnell discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Bonnell Anticipation Claim 22.

22. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3610. Bonnell discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Bonnell Anticipation Claim 24.

23. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

3611. Bonnell discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Bonnell Anticipation Claim 26. To the extent Bonnell does not by itself anticipate this claim element, Bonnell in combination with Goldman discloses this claim element. ’730 Pat. at Cl. 26; *see* Bonnell Anticipation Claim 26; Ex. A-3, claim 26; Ex. A-15, claim 26.

24. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3612. Bonnell discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Bonnell Anticipation Claim 29.

25. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3613. Bonnell discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Bonnell Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3614. Bonnell discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Bonnell Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

3615. Bonnell discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Bonnell Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *is capable of perceiving its own state; and*

3616. Bonnell discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See Bonnell Anticipation Claim 30.*

e. *is able to clone itself;*

3617. Bonnell discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See Bonnell Anticipation Claim 30.*

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3618. Bonnell discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See Bonnell Anticipation Claim 30.*

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3619. Bonnell discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See Bonnell Anticipation Claim 30.*

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3620. Bonnell discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See Bonnell Anticipation Claim 30.*

i. *including predicting a failure of a network component based on a predictive algorithm;*

3621. Bonnell discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See Bonnell Anticipation Claim 30.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3622. Bonnell discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See Bonnell Anticipation Claim 30.*

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3623. Bonnell discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See Bonnell Anticipation Claim 30.*

**26. Claim 31**

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3624. Bonnell discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See Bonnell Anticipation Claim 31.*

**27. Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3625. Bonnell discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See Bonnell Anticipation Claim 32.*

- b. *constructing a topological representation of the computer network from the information.*

3626. Bonnell discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See Bonnell Anticipation Claim 32.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

28. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3627. Bonnell discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Bonnell Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3628. Bonnell discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Bonnell Anticipation Claim 33.

29. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3629. Bonnell discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Bonnell Anticipation Claim 34.

30. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3630. Bonnell discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Bonnell Anticipation Claim 36.

**H. Boukobza in combination with Goldman and/or Hellerstein and/or**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****Kasteleijn and/or RFC 2328/Afek****1. *Motivation to Combine Boukobza with Goldman***

3631. Boukobza discloses that its agents may perform any number of policy-driven tasks, and that its agents adapt autonomously. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman.

**2. *Motivation to Combine Boukobza with Hellerstein***

3632. Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine Boukobza and Hellerstein to implement these models.

**3. *Motivation to Combine Boukobza with Kasteleijn***

3633. For example, it would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

**4. *Motivation to Combine Boukobza with RFC 2328 and/or Afek***

3634. Boukobza in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Boukobza discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boukobza and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Boukobza because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Boukobza are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Boukobza to include these elements, rendering them obvious.

5. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3635. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3636. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

3637. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1.

d. *is capable of perceiving its own state*

3638. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-16, claim 1.

e. *and is able to clone itself,*

3639. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-16, claim 1.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3640. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-16, claim 1.

3641. To the extent Natarajan does not by itself anticipate this claim element, Natarajan in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Natarajan Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-7, claim 1.

- g. *monitoring the computer network;*

3642. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3643. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-16, claim 1; Ex. A-20, claim 1.

- i. *including predicting a failure of a network component based on a prediction algorithm*

3644. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-16, claim 1; Ex. A-20, claim 1.

- j. *wherein said modeling comprises determining appropriate policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

3645. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-16, claim 1; Ex. A-20, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3646. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-16, claim 1; Ex. A-20, claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3647. Boukobza discloses this claim element. *See* Boukobza Anticipation Claim 1.

6. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3648. Boukobza discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the aBoukobza discloses this claim element. *See* Boukobza Anticipation Claim 2. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Boukobza Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-16, claim 1.

7. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3649. Boukobza discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” '730 Pat. at Cl. 3. *See* Boukobza Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *constructing a topological representation of the computer network from the information.*

3650. Boukobza discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Boukobza Anticipation Claim 3.

3651. To the extent this limitation is not expressly disclosed by Boukobza, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. Boukobza discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

8. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3652. Boukobza discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Boukobza Anticipation Claim 4.

3653. To the extent Boukobza does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Boukobza in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 4; *see* Boukobza Anticipation Claim 4; Boukobza Combination Claim 4; Ex. A-16, claim 4.

3654. Boukobza discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Boukobza Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3655. To the extent this limitation is not expressly disclosed by Boukobza, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

(<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '730 Patent. Boukobza discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3656. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Boukobza's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3657. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boukobza to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Boukobza's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

9. ***Claim 6***

a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Dijkstra Self Stabilization Algorithm.*

3658. Boukobza discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Boukobza Anticipation Claim 6.

3659. To the extent Boukobza does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Boukobza in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 6; *see* Boukobza Anticipation Claims 1, 6; Boukobza Combination Claim 1; Ex. A-16, claim 6.

3660. Boukobza discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Boukobza Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3661. To the extent this limitation is not expressly disclosed by Boukobza, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. Boukobza discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3662. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Boukobza’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3663. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boukobza to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Boukobza's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

10. ***Claim 7***

a. *A computer network, comprising:*

3664. Boukobza discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* Boukobza Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3665. Boukobza discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See* Boukobza Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3666. Boukobza discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See* Boukobza Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

3667. Boukobza discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Boukobza Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3668. Boukobza discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Boukobza Anticipation Claim 7.

f. *is able to clone itself;*

3669. Boukobza discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Boukobza Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3670. Boukobza discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Boukobza Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3671. Boukobza discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Boukobza Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3672. Boukobza discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Boukobza Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

3673. Boukobza discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Boukobza Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3674. Boukobza discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Boukobza Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3675. Boukobza discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Boukobza Anticipation Claim 7.

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3676. Boukobza discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Boukobza Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

3677. Boukobza discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Boukobza Anticipation Claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3678. Boukobza discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Boukobza Anticipation Claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3679. Boukobza discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Boukobza Anticipation Claim 13. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 13; *see* Boukobza Anticipation Claim 13; Ex. A-2, claim 13; Ex. A-16, claim 13.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3680. Boukobza discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Boukobza Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

3681. Boukobza discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Boukobza Anticipation Claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3682. Boukobza discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Boukobza Anticipation Claim 18. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 18; *see* Boukobza Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-16, claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3683. Boukobza discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Boukobza Anticipation Claim 19.

19. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3684. Boukobza discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Boukobza Anticipation Claim 21. To the extent Boukobza does not by itself anticipate this claim element, Boukobza in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 21; *see* Boukobza Anticipation Claim 21; Ex. A-2, claim 21; Ex. A-16, claim 21.

20. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism includes the graphical user interface.*

3685. Boukobza discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Boukobza Anticipation Claim 22.

21. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3686. Boukobza discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Boukobza Anticipation Claim 24.

22. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3687. Boukobza discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Boukobza Anticipation Claim 26.

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3688. Boukobza discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Boukobza Anticipation Claim 29.

24. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform a method comprising*

3689. Boukobza discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Boukobza Anticipation Claim 30.

b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3690. Boukobza discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Boukobza Anticipation Claim 30.

c. *is able to communicate with other software agents in the computer network*

3691. Boukobza discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Boukobza Anticipation Claim 30.

d. *is capable of perceiving its own state; and*

3692. Boukobza discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Boukobza Anticipation Claim 30.

e. *is able to clone itself;*

3693. Boukobza discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Boukobza Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3694. Boukobza discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* Boukobza Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

3695. Boukobza discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Boukobza Anticipation Claim 30.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3696. Boukobza discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Boukobza Anticipation Claim 30.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3697. Boukobza discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Boukobza Anticipation Claim 30.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3698. Boukobza discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Boukobza Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3699. Boukobza discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Boukobza Anticipation Claim 30.

25. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*assigned goal of the agent is expressed as a policy.*

3700. Boukobza discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Boukobza Anticipation Claim 31.

**26. Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3701. Boukobza discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Boukobza Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3702. Boukobza discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Boukobza Anticipation Claim 32.

**27. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3703. Boukobza discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Boukobza Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3704. Boukobza discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Boukobza Anticipation Claim 33.

**28. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3705. Boukobza discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Boukobza Anticipation Claim 34.

**29. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3706. Boukobza discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Boukobza Anticipation Claim 36.

**I. Hellerstein in combination with Turek and/or RFC 2328/Afek and/or Goldman and/or Kasteleijn**

**1. Motivation to Combine Hellerstein with Turek**

3707. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Hellerstein’s agents to request further policy from, for example, another agent or manager when it cannot resolve a fault or satisfy a performance target.

**2. Motivation to Combine Hellerstein with RFC 2328 and/or Afek**

3708. Hellerstein in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Hellerstein discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra’s

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Hellerstein and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Hellerstein because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Hellerstein are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Hellerstein to include these elements, rendering them obvious.

3. ***Motivation to Combine Hellerstein with Goldman***

3709. A person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer "modified" but considered "new."

4. ***Motivation to Combine Hellerstein with Kasteleijn***

3710. It would have been obvious to permit software agents to communicate amongst each other, as the MCs of Kasteleijn do, in order to permit code or functionality downloads from one agent to another, or generally exchange messages (such as multiple agents coordinating the creation of models). It would also have been obvious to permit software agents to clone themselves as the agents in Kasteleijn are cloned in order to, for example, permit an increased flexibility in the size of the network that is monitored, and the amount of modelling that can be done. Further, It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

5. ***Claim 1***

a. ***A method of managing a computer network, comprising:***

3711. Hellerstein discloses this claim element. *See Hellerstein Anticipation Claim 1.*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3712. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

- c. *is able to communicate with other software agents in the computer network;*

3713. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Hellerstein Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-20, claim 1.

- d. *is capable of perceiving its own state*

3714. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

- e. *and is able to clone itself,*

3715. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Hellerstein Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-20, claim 1.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3716. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

- g. *monitoring the computer network;*

3717. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3718. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

- i. *including predicting a failure of a network component based on a prediction algorithm*

3719. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3720. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3721. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3722. Hellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 1. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Turek discloses this claim element. '730 Pat. at Cl. 1; *see* Hellerstein Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-20, claim 1.

6. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3723. Hellerstein discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the aHellerstein discloses this claim element. *See* Hellerstein Anticipation Claim 2.

7. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3724. Hellerstein discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” '730 Pat. at Cl. 3. *See* Hellerstein Anticipation Claim 1.

- b. *constructing a topological representation of the computer network from the information.*

3725. Hellerstein discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” '730 Pat. at Cl. 3. *See* Hellerstein Anticipation Claim 3.

3726. To the extent this limitation is not disclosed by Hellerstein, it would have been obvious to combine Hellerstein with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). Hellerstein discloses agents that monitor the network by constantly checking network statistics against known thresholds. It would have been obvious at the time of invention to use path

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

information or other information from the software agents to enhance an OSPF database. As each agent collects information as to the endpoints and states of those links, the information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

8. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3727. Hellerstein discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Hellerstein Anticipation Claim 4.

3728. To the extent this limitation is not disclosed by [REF], it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

3729. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Pandya’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

9. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Dijkstra Self Stabilization Algorithm.*

3730. Hellerstein discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Hellerstein Anticipation Claim 6.

3731. To the extent Hellerstein does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Hellerstein in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* Hellerstein Anticipation Claims 1, 6; Hellerstein Combination Claim 1; Ex. A-20, claim 6.

3732. Hellerstein discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Hellerstein Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3733. To the extent this limitation is not expressly disclosed by Hellerstein, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. Hellerstein discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3734. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Hellerstein’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3735. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Hellerstein to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Hellerstein's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

10. ***Claim 7***

a. *A computer network, comprising:*

3736. Hellerstein discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* Hellerstein Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3737. Hellerstein discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See* Hellerstein Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3738. Hellerstein discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See* Hellerstein Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

3739. Hellerstein discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Hellerstein Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3740. Hellerstein discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Hellerstein Anticipation Claim 7.

f. *is able to clone itself;*

3741. Hellerstein discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Hellerstein Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3742. Hellerstein discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Hellerstein Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3743. Hellerstein discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Hellerstein Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3744. Hellerstein discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Hellerstein Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

3745. Hellerstein discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Hellerstein Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3746. Hellerstein discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Hellerstein Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3747. Hellerstein discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Hellerstein Anticipation Claim 7.

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3748. Hellerstein discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Hellerstein Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

3749. Hellerstein discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Hellerstein Anticipation Claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3750. Hellerstein discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Hellerstein Anticipation Claim 12.

3751. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 12; *see* Hellerstein Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-20, claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3752. Hellerstein discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Hellerstein Anticipation Claim 13.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3753. Hellerstein discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Hellerstein Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

3754. Hellerstein discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Hellerstein Anticipation Claim 17. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 17; *see* Hellerstein Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-20, claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3755. Hellerstein discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Hellerstein Anticipation Claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3756. Hellerstein discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Hellerstein Anticipation Claim 19. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 19; *see* Hellerstein Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-20, claim 19.

19. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3757. Hellerstein discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Hellerstein Anticipation Claim 21.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

20. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3758. Hellerstein discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Hellerstein Anticipation Claim 22.

21. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3759. Hellerstein discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Hellerstein Anticipation Claim 24.

22. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3760. Hellerstein discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Hellerstein Anticipation Claim 26. To the extent Hellerstein does not by itself anticipate this claim element, Hellerstein in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 26; *see* Hellerstein Anticipation Claim 26; Ex. A-2, claim 26; Ex. A-20, claim 26.

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3761. Hellerstein discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Hellerstein Anticipation Claim 29.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

24. *Claim 30*

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3762. Hellerstein discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See Hellerstein Anticipation Claim 30.*

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3763. Hellerstein discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Hellerstein Anticipation Claim 30.*

- c. *is able to communicate with other software agents in the computer network*

3764. Hellerstein discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See Hellerstein Anticipation Claim 30.*

- d. *is capable of perceiving its own state; and*

3765. Hellerstein discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See Hellerstein Anticipation Claim 30.*

- e. *is able to clone itself;*

3766. Hellerstein discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See Hellerstein Anticipation Claim 30.*

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3767. Hellerstein discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

for all the reasons explained above regarding claim 1, element l. *See* Hellerstein Anticipation Claim 30.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3768. Hellerstein discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Hellerstein Anticipation Claim 30.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3769. Hellerstein discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Hellerstein Anticipation Claim 30.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3770. Hellerstein discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Hellerstein Anticipation Claim 30.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3771. Hellerstein discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Hellerstein Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3772. Hellerstein discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Hellerstein Anticipation Claim 30.

**25. Claim 31**

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3773. Hellerstein discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Hellerstein Anticipation Claim 31.

**26. Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3774. Hellerstein discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Hellerstein Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3775. Hellerstein discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Hellerstein Anticipation Claim 32.

**27. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3776. Hellerstein discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Hellerstein Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the monitoring and the desired operational characteristic.*

3777. Hellerstein discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Hellerstein Anticipation Claim 33.

28. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3778. Hellerstein discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Hellerstein Anticipation Claim 34.

29. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3779. Hellerstein discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Hellerstein Anticipation Claim 36.

**J. Kasteleijn in combination with Douik and/or Douik and RFC 2328 and/or Goldman and/or Hellerstein and/or Turek and/or RFC 2328/Afek**

1. ***Motivation to Combine Kasteleijn with Douik***

3780. It would have been obvious to combine Kasteleijn and Douik because the combination is merely the use of a known technique to improve a similar device, method, or product in the same way. Kasteleijn discloses a network management system using distributed mobile agents to perform a number of network management tasks. Douik similarly discloses using agents to detect, diagnose, and correct faults by creating a qualitative simulation of device behavior. Kasteleijn in combination with Douik discloses and/or renders obvious all elements of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the '730 patent. Ex. A-2. Kasteleijn and Douik are both in the field of agent-based network management software.

**2. *Motivation to Combine Kasteleijn with Douik and RFC 2328***

3781. The claimed invention, under NetFuel's apparent interpretation of the claims as applied in its infringement contentions, is rendered obvious by the combination of Kasteleijn, Douik, and RFC 2328 (OSPF v2). The use of OSPF as an interior gateway routing protocol was well-known at the time of purported invention, and it would have been obvious at the time of invention to use OSPF with Kasteleijn and Douik. Each of Kasteleijn and Douik are in the field of software-based or defined networking.

**3. *Motivation to Combine Kasteleijn with Goldman***

3782. Kasteleijn discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the MCs), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman. A person of ordinary skill would also understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer "modified" but considered "new."

**4. *Motivation to Combine Kasteleijn with Hellerstein***

3783. Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine Kasteleijn and Hellerstein to implement these models.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****5. *Motivation to Combine Kasteleijn with Turek***

3784. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Kasteleijn's MCs to request further policy from another MC, DMF, or CCS when it cannot resolve a fault or satisfy a performance target. Further, a person of ordinary skill in the art would understand that network management tools may benefit from a GUI.

**6. *Motivation to Combine Kasteleijn with RFC 2328 and/or Afek***

3785. Kasteleijn in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Kasteleijn discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Kasteleijn and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Kasteleijn because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Kasteleijn are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Kasteleijn to include these elements, rendering them obvious.

**7. *Claim 1***

a. *A method of managing a computer network, comprising:*

3786. Kasteleijn discloses this claim element. *See Kasteleijn Anticipation Claim 1.*

b. *assigning a goal to a software [agent], wherein the software agent*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*has its own runtime environment*

3787. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

3788. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1.

d. *is capable of perceiving its own state*

3789. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1.

e. *and is able to clone itself,*

3790. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3791. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1.

g. *monitoring the computer network;*

3792. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3793. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1; Ex. A-20, claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

3794. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1; Ex. A-20, claim 1.

j. *wherein said modeling comprises determining appropriate policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

3795. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1; Ex. A-20, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3796. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-3, claim 1; Ex. A-20, claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3797. Kasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 1. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Turek discloses this claim element. '730 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-2, claim 1.

8. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3798. Kasteleijn discloses claim 2, "[t]he method of claim 2, wherein the assigned goal of the aKasteleijn discloses this claim element. *See* Kasteleijn Anticipation Claim 2.

9. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3799. Kasteleijn discloses this element of claim 3, "[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. *See* Kasteleijn Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *constructing a topological representation of the computer network from the information.*

3800. Kasteleijn discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Kasteleijn Anticipation Claim 3. To the extent this limitation is not disclosed by Kasteleijn, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). Kasteleijn discloses mobile agents traversing a network and collecting information for reporting back to a central manager. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

10. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3801. Kasteleijn discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Kasteleijn Anticipation Claim 4.

3802. To the extent this limitation is not disclosed by Kasteleijn, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

3803. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Pandya’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

11. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3804. Kasteleijn discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Kasteleijn Anticipation Claim 6.

3805. To the extent Kasteleijn does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Kasteleijn in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* Kasteleijn Anticipation Claims 1, 6; Kasteleijn Combination Claim 1; Ex. A-2, claim 6.

3806. Kasteleijn discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Kasteleijn Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3807. To the extent this limitation is not expressly disclosed by Kasteleijn, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. Kasteleijn discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3808. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Kasteleijn's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3809. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Kasteleijn to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Kasteleijn's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

12. **Claim 7**

a. *A computer network, comprising:*

3810. Kasteleijn discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* Kasteleijn Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3811. Kasteleijn discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Kasteleijn Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3812. Kasteleijn discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Kasteleijn Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

3813. Kasteleijn discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Kasteleijn Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3814. Kasteleijn discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Kasteleijn Anticipation Claim 7.

f. *is able to clone itself;*

3815. Kasteleijn discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Kasteleijn Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3816. Kasteleijn discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Kasteleijn Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3817. Kasteleijn discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Kasteleijn Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3818. Kasteleijn discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Kasteleijn Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

3819. Kasteleijn discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Kasteleijn Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3820. Kasteleijn discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Kasteleijn Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3821. Kasteleijn discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Kasteleijn Anticipation Claim 7.

13. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*an operational characteristic of the network.*

3822. Kasteleijn discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Kasteleijn Anticipation Claim 10.

14. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

3823. Kasteleijn discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Kasteleijn Anticipation Claim 11.

15. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3824. Kasteleijn discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Kasteleijn Anticipation Claim 12.

16. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3825. Kasteleijn discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Kasteleijn Anticipation Claim 13.

17. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*for a particular network component.*

3826. Kasteleijn discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Kasteleijn Anticipation Claim 16.

18. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

3827. Kasteleijn discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Kasteleijn Anticipation Claim 17.

19. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3828. Kasteleijn discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Kasteleijn Anticipation Claim 18.

20. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3829. Kasteleijn discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Kasteleijn Anticipation Claim 19.

21. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3830. Kasteleijn discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Kasteleijn Anticipation Claim 21.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**22. Claim 22**

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3831. Kasteleijn discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Kasteleijn Anticipation Claim 22. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Turek discloses this claim element. ’730 Pat. at Cl. 22; *see* Kasteleijn Anticipation Claim 22; Ex. A-1, claim 22; Ex. A-2, claim 22.

**23. Claim 24**

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3832. Kasteleijn discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Kasteleijn Anticipation Claim 24.

**24. Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3833. Kasteleijn discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Kasteleijn Anticipation Claim 26. To the extent Kasteleijn does not by itself anticipate this claim element, Kasteleijn in combination with Goldman discloses this claim element. ’730 Pat. at Cl. 26; *see* Kasteleijn Anticipation Claim 26; Ex. A-2, claim 26; Ex. A-3, claim 26.

**25. Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3834. Kasteleijn discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

without having to request further policy.” ’730 Pat. at Cl. 29. *See Kasteleijn Anticipation Claim 29.*

26. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3835. Kasteleijn discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See Kasteleijn Anticipation Claim 30.*

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3836. Kasteleijn discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Kasteleijn Anticipation Claim 30.*

- c. *is able to communicate with other software agents in the computer network*

3837. Kasteleijn discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See Kasteleijn Anticipation Claim 30.*

- d. *is capable of perceiving its own state; and*

3838. Kasteleijn discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See Kasteleijn Anticipation Claim 30.*

- e. *is able to clone itself;*

3839. Kasteleijn discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See Kasteleijn Anticipation Claim 30.*

- f. *and comprises an autonomous agent operable to request further*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy when it lacks an ability to perform the predefined task*

3840. Kasteleijn discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l. *See* Kasteleijn Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3841. Kasteleijn discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Kasteleijn Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3842. Kasteleijn discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Kasteleijn Anticipation Claim 30.

i. *including predicting a failure of a network component based on a predictive algorithm;*

3843. Kasteleijn discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Kasteleijn Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3844. Kasteleijn discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Kasteleijn Anticipation Claim 30.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

3845. Kasteleijn discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Kasteleijn Anticipation Claim 30.

27. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3846. Kasteleijn discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Kasteleijn Anticipation Claim 31.

28. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3847. Kasteleijn discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Kasteleijn Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3848. Kasteleijn discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Kasteleijn Anticipation Claim 32.

29. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3849. Kasteleijn discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Kasteleijn Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3850. Kasteleijn discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Kasteleijn Anticipation Claim 33.

30. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3851. Kasteleijn discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Kasteleijn Anticipation Claim 34.

31. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3852. Kasteleijn discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Kasteleijn Anticipation Claim 36.

**K. da Rocha in combination with Kasteleijn and/or Goldman and/or Turek and/or RFC 2328/Afek**

1. ***Motivation to Combine Da Rocha with Kasteleijn***

3853. It would have been obvious to permit software agents to communicate amongst each other, as the MCs of Kasteleijn do, in order to permit code or functionality downloads from one agent to another, or generally exchange messages (such as informing other agents that they may stop processing because the fault has been located). Further, it would have been obvious to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

permit software agents to clone themselves as the agents in Kasteleijn are cloned in order to permit an increased flexibility in the size of the network that is monitored. It would also have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

**2. *Motivation to Combine Da Rocha with Goldman***

3854. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when executed by da Rocha's agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* additional code or agents), as taught by Goldman. Further, a person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer "modified" but considered "new."

**3. *Motivation to Combine Da Rocha with Turek***

3855. It would have been obvious at the time of invention to one of ordinary skill in the art to enable da Rocha's agents to request further policy from another agent, manager, or network administrator when it cannot resolve a fault or satisfy a performance target.

**4. *Motivation to Combine Da Rocha with RFC 2328 and/or Afek***

3856. Da Rocha in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Da Rocha discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame,

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

would have motivated one having ordinary skill in the art to combine the teachings of Da Rocha and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Da Rocha because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Da Rocha are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Da Rocha to include these elements, rendering them obvious.

5. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3857. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3858. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

3859. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-6, claim 1.

d. *is capable of perceiving its own state*

3860. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1.

e. *and is able to clone itself,*

3861. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-6, claim 1.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

3862. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1.

- g. *monitoring the computer network;*

3863. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3864. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Goldman discloses this claim element. '730 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1.

- i. *including predicting a failure of a network component based on a prediction algorithm*

3865. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Goldman discloses this claim element. '730 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3866. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Goldman discloses this claim element. '730 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3867. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Goldman discloses this claim element. '730 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3868. da Rocha discloses this claim element. *See* da Rocha Anticipation Claim 1. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Turek discloses this claim element. '730 Pat. at Cl. 1; *see* da Rocha Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1.

6. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3869. da Rocha discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the ada Rocha discloses this claim element. *See* da Rocha Anticipation Claim 2.

7. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3870. da Rocha discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* da Rocha Anticipation Claim 1.

- b. *constructing a topological representation of the computer network from the information.*

3871. da Rocha discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* da Rocha Anticipation Claim 3.

3872. To the extent this limitation is not disclosed by da Rocha, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). da Rocha discloses agents collecting information at a network node for reporting back to a central manager. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. Each agent may collect information as to

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the states of network links. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

8. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3873. da Rocha discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* da Rocha Anticipation Claim 4.

3874. To the extent this limitation is not disclosed by da Rocha, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

3875. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Pandya’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

9. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Dijkstra Self Stabilization Algorithm.*

3876. da Rocha discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* da Rocha Anticipation Claim 6.

3877. To the extent da Rocha does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, da Rocha in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* da Rocha Anticipation Claims 1, 6; da Rocha Combination Claim 1; Ex. A-6, claim 6.

3878. da Rocha discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* da Rocha Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3879. To the extent this limitation is not expressly disclosed by da Rocha, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. da Rocha discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3880. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in da Rocha’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3881. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify da Rocha to create a corrective policy using Dijkstra's Self Stabilization Algorithm in da Rocha's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

10. ***Claim 7***

a. *A computer network, comprising:*

3882. da Rocha discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* da Rocha Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3883. da Rocha discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See* da Rocha Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3884. da Rocha discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See* da Rocha Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

3885. da Rocha discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* da Rocha Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3886. da Rocha discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* da Rocha Anticipation Claim 7.

f. *is able to clone itself;*

3887. da Rocha discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* da Rocha Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3888. da Rocha discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* da Rocha Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3889. da Rocha discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* da Rocha Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3890. da Rocha discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* da Rocha Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

3891. da Rocha discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* da Rocha Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3892. da Rocha discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* da Rocha Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3893. da Rocha discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* da Rocha Anticipation Claim 7.

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3894. da Rocha discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* da Rocha Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

3895. da Rocha discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* da Rocha Anticipation Claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3896. da Rocha discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* da Rocha Anticipation Claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3897. da Rocha discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* da Rocha Anticipation Claim 13. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 12; *see* da Rocha Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-6, claim 12.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3898. da Rocha discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* da Rocha Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

3899. da Rocha discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* da Rocha Anticipation Claim 17. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 17; *see* da Rocha Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-6, claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3900. da Rocha discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* da Rocha Anticipation Claim 18. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 18; *see* da Rocha Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-6, claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3901. da Rocha discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* da Rocha Anticipation Claim 19. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 19; *see* da Rocha Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-6, claim 19.

19. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*database to store the policy for the agent.*

3902. da Rocha discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* da Rocha Anticipation Claim 21.

20. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3903. da Rocha discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* da Rocha Anticipation Claim 22.

21. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3904. da Rocha discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* da Rocha Anticipation Claim 24.

22. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3905. da Rocha discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* da Rocha Anticipation Claim 26. To the extent da Rocha does not by itself anticipate this claim element, da Rocha in combination with Goldman discloses this claim element. ’730 Pat. at Cl. 26; *see* da Rocha Anticipation Claim 26; Ex. A-3, claim 26; Ex. A-6, claim 26.

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

3906. da Rocha discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* da Rocha Anticipation Claim 29.

24. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

3907. da Rocha discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* da Rocha Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3908. da Rocha discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* da Rocha Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

3909. da Rocha discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* da Rocha Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

3910. da Rocha discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* da Rocha Anticipation Claim 30.

- e. *is able to clone itself;*

3911. da Rocha discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* da Rocha Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3912. da Rocha discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l. *See* da Rocha Anticipation Claim 30.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

3913. da Rocha discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* da Rocha Anticipation Claim 30.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3914. da Rocha discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* da Rocha Anticipation Claim 30.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3915. da Rocha discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* da Rocha Anticipation Claim 30.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3916. da Rocha discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* da Rocha Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

3917. da Rocha discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* da Rocha Anticipation Claim 30.

**25. Claim 31**

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

3918. da Rocha discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* da Rocha Anticipation Claim 31.

**26. Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3919. da Rocha discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* da Rocha Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3920. da Rocha discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* da Rocha Anticipation Claim 32.

**27. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3921. da Rocha discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See da Rocha Anticipation Claim 33.*

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3922. da Rocha discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See da Rocha Anticipation Claim 33.*

28. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3923. da Rocha discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See da Rocha Anticipation Claim 34.*

29. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3924. da Rocha discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See da Rocha Anticipation Claim 36.*

**L. INCA in combination with Kasteleijn and/or Goldman and/or Hellerstein and/or RFC 2328/Afek and/or Turek**

1. ***Motivation to Combine INCA with Kasteleijn***

3925. Caching an agent’s state and then restoring it implies the ability to clone the agent. Furthermore, this limitation is rendered obvious in combination with Kasteleijn. *See Ex. A-2, limitation 1.4.* It would have been obvious to permit software agents to clone themselves as the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agents in Kasteleijn are cloned in order to, for example, permit an increased flexibility in the size of the network that is monitored.

**2. *Motivation to Combine INCA with Goldman***

3926. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to INCA's agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* additional code or agents), as taught by Goldman. A person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer "modified" but considered "new."

**3. *Motivation to Combine INCA with Hellerstein***

3927. Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine INCA and Hellerstein to implement these models.

**4. *Motivation to Combine INCA with RFC 2328 and/or Afek***

3928. INCA in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. INCA discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of INCA and

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in INCA because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and INCA are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or INCA to include these elements, rendering them obvious.

5. ***Motivation to Combine INCA with Turek***

3929. A person of ordinary skill in the art would understand that network management tools may benefit from a GUI.

6. ***Claim 1***

a. *A method of managing a computer network, comprising:*

3930. INCA discloses this claim element. *See* INCA Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

3931. INCA discloses this claim element. *See* INCA Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

3932. INCA discloses this claim element. *See* INCA Anticipation Claim 1.

d. *is capable of perceiving its own state*

3933. INCA discloses this claim element. *See* INCA Anticipation Claim 1.

e. *and is able to clone itself,*

3934. INCA discloses this claim element. *See* INCA Anticipation Claim 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-13, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent; and*

3935. INCA discloses this claim element. *See* INCA Anticipation Claim 1.

g. *monitoring the computer network;*

3936. INCA discloses this claim element. *See* INCA Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

3937. INCA discloses this claim element. *See* INCA Anticipation Claim 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. A-13, claim 1; Ex. A-20, claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

3938. INCA discloses this claim element. *See* INCA Anticipation Claim 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-13, claim 1; Ex. A-20, claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3939. INCA discloses this claim element. *See* INCA Anticipation Claim 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-13, claim 1; Ex. A-20, claim 1.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

3940. INCA discloses this claim element. *See* INCA Anticipation Claim 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Goldman and/or Hellerstein discloses this claim element. '730 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-13, claim 1; Ex. A-20, claim 1.

l. *wherein the software agent comprises an autonomous agent*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*operable to request further policy when it lacks an ability to perform the predefined task.*

3941. INCA discloses this claim element. *See* INCA Anticipation Claim 1. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Turek discloses this claim element. '730 Pat. at Cl. 1; *see* INCA Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-13, claim 1.

7. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

3942. INCA discloses claim 2, “[t]he method of claim 2, wherein the assigned goal of the aINCA discloses this claim element. *See* INCA Anticipation Claim 2.

8. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

3943. INCA discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” '730 Pat. at Cl. 3. *See* INCA Anticipation Claim 1.

- b. *constructing a topological representation of the computer network from the information.*

3944. INCA discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” '730 Pat. at Cl. 3. *See* INCA Anticipation Claim 3.

3945. To the extent this limitation is not disclosed by INCA, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). INCA discloses mobile agents traversing a network and collecting information for reporting back to a central manager. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

those links. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

9. **Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

3946. INCA discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* INCA Anticipation Claim 4.

3947. To the extent this limitation is not disclosed by INCA, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

3948. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Pandya’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

10. **Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Dijkstra Self Stabilization Algorithm.*

3949. INCA discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* INCA Anticipation Claim 6.

3950. To the extent INCA does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, INCA in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* INCA Anticipation Claims 1, 6; INCA Combination Claim 1; Ex. A-13, claim 6.

3951. INCA discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* INCA Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

3952. To the extent this limitation is not expressly disclosed by INCA, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. INCA discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

3953. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in INCA’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

3954. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify INCA to create a corrective policy using Dijkstra's Self Stabilization Algorithm in INCA's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

11. ***Claim 7***

a. *A computer network, comprising:*

3955. INCA discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* INCA Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

3956. INCA discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See* INCA Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

3957. INCA discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See* INCA Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

3958. INCA discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* INCA Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

3959. INCA discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* INCA Anticipation Claim 7.

f. *is able to clone itself;*

3960. INCA discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* INCA Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

3961. INCA discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* INCA Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

3962. INCA discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* INCA Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

3963. INCA discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* INCA Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

3964. INCA discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* INCA Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

3965. INCA discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* INCA Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

3966. INCA discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* INCA Anticipation Claim 7.

12. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

3967. INCA discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* INCA Anticipation Claim 10.

13. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

3968. INCA discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* INCA Anticipation Claim 11.

14. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

3969. INCA discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* INCA Anticipation Claim 12. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 12; *see* INCA Anticipation Claim 12; Ex. A-2, claim 13; Ex. A-13, claim 13.

15. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

3970. INCA discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* INCA Anticipation Claim 13.

16. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

3971. INCA discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* INCA Anticipation Claim 16.

17. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

3972. INCA discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* INCA Anticipation Claim 17.

18. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

3973. INCA discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* INCA Anticipation Claim 18.

19. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

3974. INCA discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* INCA Anticipation Claim 19.

20. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

3975. INCA discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* INCA Anticipation Claim 21.

21. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

3976. INCA discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* INCA Anticipation Claim 22.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

3977. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Turek discloses this claim element. '730 Pat. at Cl. 22; *see* INCA Anticipation Claim 22; Ex. A-1, claim 22; Ex. A-13, claim 22.

**22. Claim 24**

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

3978. INCA discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* INCA Anticipation Claim 24.

**23. Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

3979. INCA discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” '730 Pat. at Cl. 26. *See* INCA Anticipation Claim 26. To the extent INCA does not by itself anticipate this claim element, INCA in combination with Goldman discloses this claim element. '730 Pat. at Cl. 26; *see* INCA Anticipation Claim 26; Ex. A-3, claim 26; Ex. A-13, claim 26.

**24. Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

3980. INCA discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” '730 Pat. at Cl. 29. *See* INCA Anticipation Claim 29.

**25. Claim 30**

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform a method comprising*

3981. INCA discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* INCA Anticipation Claim 30.

b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

3982. INCA discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* INCA Anticipation Claim 30.

c. *is able to communicate with other software agents in the computer network*

3983. INCA discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* INCA Anticipation Claim 30.

d. *is capable of perceiving its own state; and*

3984. INCA discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* INCA Anticipation Claim 30.

e. *is able to clone itself;*

3985. INCA discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* INCA Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

3986. INCA discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* INCA Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

3987. INCA discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* INCA Anticipation Claim 30.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

3988. INCA discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* INCA Anticipation Claim 30.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

3989. INCA discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* INCA Anticipation Claim 30.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

3990. INCA discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* INCA Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

3991. INCA discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* INCA Anticipation Claim 30.

26. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*assigned goal of the agent is expressed as a policy.*

3992. INCA discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* INCA Anticipation Claim 31.

27. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

3993. INCA discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* INCA Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

3994. INCA discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* INCA Anticipation Claim 32.

28. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

3995. INCA discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* INCA Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

3996. INCA discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

all the reasons explained above regarding claim 1, elements h through k. *See* INCA Anticipation Claim 33.

**29. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

3997. INCA discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* INCA Anticipation Claim 34.

**30. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

3998. INCA discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* INCA Anticipation Claim 36.

**M. Boudaoud in combination with Kasteleijn and/or RFC 2328/Afek and/or Goldman**

**1. Motivation to Combine Boudaoud with Kasteleijn**

3999. It would have been obvious to permit software agents to clone themselves as the agents in Kasteleijn are cloned, for example, in order to permit an increased flexibility in the size of the network that is monitored. Further, It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

**2. Motivation to Combine Boudaoud with RFC 2328 and/or Afek**

4000. Boudaoud in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Boudaoud

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Boudaoud and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Boudaoud because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Boudaoud are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Boudaoud to include these elements, rendering them obvious.

3. ***Motivation to Combine Boudaoud with Goldman***

4001. A person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer "modified" but considered "new."

4. ***Claim 1***

a. *A method of managing a computer network, comprising:*

4002. Boudaoud discloses this claim element. *See Boudaoud Anticipation Claim 1.*

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

4003. Boudaoud discloses this claim element. *See Boudaoud Anticipation Claim 1.*

c. *is able to communicate with other software agents in the computer network;*

4004. Boudaoud discloses this claim element. *See Boudaoud Anticipation Claim 1.*

d. *is capable of perceiving its own state*

4005. Boudaoud discloses this claim element. *See Boudaoud Anticipation Claim 1.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *and is able to clone itself,*

4006. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Boudaoud Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-14, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4007. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

g. *monitoring the computer network;*

4008. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4009. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

4010. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4011. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

4012. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4013. Boudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 1.

**5. Claim 2**

a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4014. Boudaoud discloses claim 2, "[t]he method of claim 2, wherein the assigned goal of the aBoudaoud discloses this claim element. *See* Boudaoud Anticipation Claim 2.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**6. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4015. Boudaoud discloses this element of claim 3, “[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* Boudaoud Anticipation Claim 1.

- b. *constructing a topological representation of the computer network from the information.*

4016. Boudaoud discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Boudaoud Anticipation Claim 3.

4017. To the extent this limitation is not disclosed by Boudaoud ’98, it would have been obvious to combine Boudaoud ’98 with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). Boudaoud ’98 discloses mobile agents that traverse the network while gathering information. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

**7. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4018. Boudaoud discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Boudaoud Anticipation Claim 4.

4019. To the extent this limitation is not disclosed by Boudaoud, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4020. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Pandya's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

8. **Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4021. Boudaoud discloses claim 6, "[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm." '730 Pat. at Cl. 6. *See* Boudaoud Anticipation Claim 6.

4022. To the extent Boudaoud does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Boudaoud in combination with RFC 2328 and/or Afek discloses this claim element. '730 Pat. at Cl. 1; *see* Boudaoud Anticipation Claims 1, 6; Boudaoud Combination Claim 1; Ex. A-14, claim 6.

4023. Boudaoud discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Boudaoud Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4024. To the extent this limitation is not expressly disclosed by Boudaoud, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '730 Patent. Boudaoud discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

4025. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Boudaoud's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

4026. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Boudaoud to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Boudaoud's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

9. *Claim 7*

a. *A computer network, comprising:*

4027. Boudaoud discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* Boudaoud Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4028. Boudaoud discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Boudaoud Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

4029. Boudaoud discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Boudaoud Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

4030. Boudaoud discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Boudaoud Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

4031. Boudaoud discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Boudaoud Anticipation Claim 7.

f. *is able to clone itself;*

4032. Boudaoud discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Boudaoud Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

4033. Boudaoud discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Boudaoud Anticipation Claim 7.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4034. Boudaoud discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Boudaoud Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

4035. Boudaoud discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Boudaoud Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

4036. Boudaoud discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Boudaoud Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4037. Boudaoud discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Boudaoud Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

4038. Boudaoud discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Boudaoud Anticipation Claim 7.

10. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4039. Boudaoud discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Boudaoud Anticipation Claim 10.

11. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4040. Boudaoud discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Boudaoud Anticipation Claim 11.

12. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4041. Boudaoud discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Boudaoud Anticipation Claim 12.

13. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications protocol encrypts a payload of a data packet.*

4042. Boudaoud discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Boudaoud Anticipation Claim 13. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 12; *see* Boudaoud Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-14, claim 12.

14. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

4043. Boudaoud discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Boudaoud Anticipation Claim 16.

15. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4044. Boudaoud discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Boudaoud Anticipation Claim 17.

16. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4045. Boudaoud discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Boudaoud Anticipation Claim 18.

17. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*environment.*

4046. Boudaoud discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Boudaoud Anticipation Claim 19. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 19; *see* Boudaoud Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-14, claim 19.

18. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4047. Boudaoud discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Boudaoud Anticipation Claim 21.

19. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4048. Boudaoud discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Boudaoud Anticipation Claim 22.

20. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4049. Boudaoud discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Boudaoud Anticipation Claim 24.

21. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

4050. Boudaoud discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Boudaoud Anticipation Claim 26. To the extent Boudaoud does not by itself anticipate this claim element, Boudaoud in combination with Goldman discloses this claim element. ’730 Pat. at Cl. 26; *see* Boudaoud Anticipation Claim 26; Ex. A-3, claim 26; Ex. A-14, claim 26.

22. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

4051. Boudaoud discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Boudaoud Anticipation Claim 29.

23. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4052. Boudaoud discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Boudaoud Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4053. Boudaoud discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Boudaoud Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

4054. Boudaoud discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Boudaoud Anticipation Claim 30.

d. *is capable of perceiving its own state; and*

4055. Boudaoud discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Boudaoud Anticipation Claim 30.

e. *is able to clone itself;*

4056. Boudaoud discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Boudaoud Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4057. Boudaoud discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* Boudaoud Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4058. Boudaoud discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See* Boudaoud Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4059. Boudaoud discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Boudaoud Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- i. *including predicting a failure of a network component based on a predictive algorithm;*

4060. Boudaoud discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See Boudaoud Anticipation Claim 30.*

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4061. Boudaoud discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See Boudaoud Anticipation Claim 30.*

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

4062. Boudaoud discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See Boudaoud Anticipation Claim 30.*

24. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

4063. Boudaoud discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See Boudaoud Anticipation Claim 31.*

25. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4064. Boudaoud discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See Boudaoud Anticipation Claim 32.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *constructing a topological representation of the computer network from the information.*

4065. Boudaoud discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Boudaoud Anticipation Claim 32.

**26. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4066. Boudaoud discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Boudaoud Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4067. Boudaoud discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Boudaoud Anticipation Claim 33.

**27. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4068. Boudaoud discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Boudaoud Anticipation Claim 34.

**28. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Algorithm.*

4069. Boudaoud discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Boudaoud Anticipation Claim 36.

**N. Yoshihara in combination with Kasteleijn and/or Goldman and/or Castelli and/or Turek and/or Natarajan and/or RFC 2328/Afek**

**1. *Motivation to Combine Yoshihara with Kasteleijn***

4070. It would have been obvious at the time of invention to one of ordinary skill in the art to utilize the secure communications channel of Kasteleijn to effect.

**2. *Motivation to Combine Yoshihara with Goldman***

4071. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to Yoshihara’s agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* additional code or agents), as taught by Goldman.

**3. *Motivation to Combine Yoshihara with Castelli***

4072. It would have been obvious to perform policy adaptation based on the likelihood of failure of a component, such as the processor aging or load balancing described in Castelli. *See* Ex. A-12 at limitation 1.8.

**4. *Motivation to Combine Yoshihara with Turek***

4073. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Yoshihara’s managed systems to request further policy code when it cannot resolve a fault or satisfy a performance target on its own.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****5. *Motivation to Combine Yoshihara with Natarajan***

4074. It would have been obvious at the time of invention to a skilled artisan to employ Yoshihara's managed systems to request further policy from a policy-providing entity when it cannot solve an issue or satisfy a performance target.

**6. *Motivation to Combine Yoshihara with RFC 2328 and/or Afek***

4075. Yoshihara in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Yoshihara discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Yoshihara and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Yoshihara because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Yoshihara are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Yoshihara to include these elements, rendering them obvious.

**7. *Claim 1*****a. *A method of managing a computer network, comprising:***

4076. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

**b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment***

4077. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. *is able to communicate with other software agents in the computer network;*

4078. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-21, claim 1.

- d. *is capable of perceiving its own state*

4079. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

- e. *and is able to clone itself,*

4080. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Kasteleijn discloses this claim element. '730 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-21, claim 1.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4081. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

- g. *monitoring the computer network;*

4082. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4083. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

- i. *including predicting a failure of a network component based on a prediction algorithm*

4084. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Goldman and/or Castelli discloses this claim element. '730 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-12, claim 1; Ex. A-21, claim 1.

- j. *wherein said modeling comprises determining appropriate policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

4085. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Goldman and/or Castelli discloses this claim element. '730 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-12, claim 1; Ex. A-21, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

4086. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4087. Yoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 1. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Turek and/or Natarajan discloses this claim element. '730 Pat. at Cl. 1; *see* Yoshihara Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-7, claim 1; Ex. A-21, claim 1.

8. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4088. Yoshihara discloses claim 2, "[t]he method of claim 2, wherein the assigned goal of the aYoshihara discloses this claim element. *See* Yoshihara Anticipation Claim 2.

9. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4089. Yoshihara discloses this element of claim 3, "[t]he method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. *See* Yoshihara Anticipation Claim 1.

- b. *constructing a topological representation of the computer network*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

4090. Yoshihara discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Yoshihara Anticipation Claim 3.

4091. To the extent this limitation is not disclosed by Yoshihara, it would have been obvious to combine Yoshihara with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). Yoshihara using distributed agents to monitor operational parameters of the network. It would have been obvious at the time of invention to use path information or other information from the software agents to enhance an OSPF database. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

10. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4092. Yoshihara discloses claim 4, “[t]he method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Yoshihara Anticipation Claim 4.

4093. To the extent this limitation is not disclosed by Yoshihara, it would have been obvious to combine Yoshihara with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). NetFuel contends that initiating a policy that blocks OSPF packets satisfies this limitation. It would have been obvious at the time of invention to apply quality of service or bandwidth metering as discussed by Yoshihara to OSPF packets.

4094. Furthermore, modeling using numerical models such as the Dijkstra self-stabilization algorithm were well known at the time of invention. It would have been obvious, and merely a design choice, to create a policy adaptation algorithm in Yoshihara that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek.

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

11. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4095. Yoshihara discloses claim 6, “[t]he method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Yoshihara Anticipation Claim 6.

4096. To the extent Yoshihara does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Yoshihara in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Pat. at Cl. 1; *see* Yoshihara Anticipation Claims 1, 6; Yoshihara Combination Claim 1; Ex. A-21, claim 6.

4097. Yoshihara discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Yoshihara Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

4098. To the extent this limitation is not expressly disclosed by Yoshihara, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. Yoshihara discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4099. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Yoshihara's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

*In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at pp. 200-204.

4100. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Yoshihara to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Yoshihara's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

12. **Claim 7**

a. *A computer network, comprising:*

4101. Yoshihara discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. See Yoshihara Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4102. Yoshihara discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Yoshihara Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

4103. Yoshihara discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Yoshihara Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

4104. Yoshihara discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Yoshihara Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

4105. Yoshihara discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Yoshihara Anticipation Claim 7.

f. *is able to clone itself;*

4106. Yoshihara discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Yoshihara Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

4107. Yoshihara discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Yoshihara Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4108. Yoshihara discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Yoshihara Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

4109. Yoshihara discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Yoshihara Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

4110. Yoshihara discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Yoshihara Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4111. Yoshihara discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Yoshihara Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4112. Yoshihara discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Yoshihara Anticipation Claim 7.

13. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*an operational characteristic of the network.*

4113. Yoshihara discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Yoshihara Anticipation Claim 10.

14. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4114. Yoshihara discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Yoshihara Anticipation Claim 11.

15. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4115. Yoshihara discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Yoshihara Anticipation Claim 12.

16. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

4116. Yoshihara discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Yoshihara Anticipation Claim 13. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 13; *see* Yoshihara Anticipation Claim 13; Ex. A-2, claim 13; Ex. A-21, claim 13.

17. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*for a particular network component.*

4117. Yoshihara discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Yoshihara Anticipation Claim 16.

18. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4118. Yoshihara discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Yoshihara Anticipation Claim 17. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 17; *see* Yoshihara Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-21, claim 17.

19. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4119. Yoshihara discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Yoshihara Anticipation Claim 18. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Kasteleijn discloses this claim element. ’730 Pat. at Cl. 18; *see* Yoshihara Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-21, claim 18.

20. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4120. Yoshihara discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Yoshihara Anticipation Claim 19. To the extent Yoshihara does not by itself anticipate this claim element, Yoshihara in combination with Kasteleijn discloses this claim

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

element. '730 Pat. at Cl. 19; *see* Yoshihara Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-21, claim 19.

**21. Claim 21**

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4121. Yoshihara discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Yoshihara Anticipation Claim 21.

**22. Claim 22**

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4122. Yoshihara discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Yoshihara Anticipation Claim 22.

**23. Claim 24**

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4123. Yoshihara discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Yoshihara Anticipation Claim 24.

**24. Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

4124. Yoshihara discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Yoshihara Anticipation Claim 26.

**25. Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

4125. Yoshihara discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Yoshihara Anticipation Claim 29.

26. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4126. Yoshihara discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Yoshihara Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4127. Yoshihara discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Yoshihara Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

4128. Yoshihara discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Yoshihara Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

4129. Yoshihara discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Yoshihara Anticipation Claim 30.

- e. *is able to clone itself;*

4130. Yoshihara discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Yoshihara Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4131. Yoshihara discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l. *See* Yoshihara Anticipation Claim 30.

- g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4132. Yoshihara discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Yoshihara Anticipation Claim 30.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4133. Yoshihara discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Yoshihara Anticipation Claim 30.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

4134. Yoshihara discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Yoshihara Anticipation Claim 30.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4135. Yoshihara discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Yoshihara Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

4136. Yoshihara discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Yoshihara Anticipation Claim 30.

27. ***Claim 31***

- a. *The machine-readable storage medium of claim 30, wherein the assigned goal of the agent is expressed as a policy.*

4137. Yoshihara discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Yoshihara Anticipation Claim 31.

28. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4138. Yoshihara discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Yoshihara Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

4139. Yoshihara discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Yoshihara Anticipation Claim 32.

29. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4140. Yoshihara discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Yoshihara Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4141. Yoshihara discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Yoshihara Anticipation Claim 33.

30. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4142. Yoshihara discloses this element of claim 30, “[t]he machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Yoshihara Anticipation Claim 34.

31. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4143. Yoshihara discloses this element of claim 30, “[t]he machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Yoshihara Anticipation Claim 36.

**O. Cisco NetRanger in combination with Kasteleijn, and/or Goldman, and/or de Rocha, and/or RFC 2328**

1. ***Motivation to Combine Cisco NetRanger with Kasteleijn***

4144. It would have been obvious to one of ordinary skill in the art to modify the agents of NetRanger to include the capability to clone themselves, for faster provisioning of new sensors, or to permit agent mobility, as described in Kasteleijn.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. Motivation to combine Cisco NetRanger with Goldman**

4145. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that will not result in failure), where modeling includes predicting failure of a network component (e.g. faults, inconsistencies or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new shun or other traffic settings), as taught by Goldman. *See* Ex. A-3, limitation 1.7.

**3. Motivation to combine Cisco NetRanger with de Rocha**

4146. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. faults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new shun or other traffic settings), as taught by da Rocha. *See* Ex. A-6, limitation 1.7.

**4. Motivation to combine Cisco NetRanger with RFC 2328**

4147. Cisco NetRanger in combination with RFC 2328 renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco NetRanger discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco NetRanger and RFC 2328. For example, a person of ordinary

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

skill in the art would have been motivated to implement RFC 2328's self-stabilizing algorithms in Cisco NetRanger because the RFC 2328 techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and Cisco NetRanger are found to not include each element of the asserted claims of the '730 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Cisco NetRanger to include these elements, rendering them obvious.

5. ***Claim 1***

a. *A method of managing a computer network, comprising:*

4148. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

4149. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

4150. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

d. *is capable of perceiving its own state*

4151. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

e. *and is able to clone itself,*

4152. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* NetRanger Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-8, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4153. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

g. *monitoring the computer network;*

4154. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4155. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Goldman and/or de Rocha discloses this claim element. '730 Patent claim 1; *see* NetRanger Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1; Ex. A-8, claim 1.

- i. *including predicting a failure of a network component based on a prediction algorithm*

4156. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4157. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Goldman and/or da Rocha discloses this claim element. '730 Patent claim 1; *see* NetRanger Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1; Ex. A-8, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

4158. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1. To the extent NetRanger does not by itself anticipate this claim element, NetRanger in combination with Goldman and/or de Rocha discloses this claim element. '730 Patent claim 1; *see* NetRanger Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1; Ex. A-8, claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4159. NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 1.

**6. Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4160. NetRanger discloses claim 2, "The method of claim 2, wherein the assigned goal of the NetRanger discloses this claim element. *See* NetRanger Anticipation Claim 2.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

7. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4161. NetRanger discloses this element of claim 3, “The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* NetRanger Anticipation Claim 3.

- b. *constructing a topological representation of the computer network from the information.*

4162. NetRanger discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* NetRanger Anticipation Claim 3.

4163. To the extent this limitation is not disclosed in NetRanger, it would have been obvious to combine NetRanger with the OSFP routing protocol (RFC 2328). Under NetFuel’s interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. NetRanger monitors ICMP messages that change the routing table, and it would have been obvious to use NetRanger on a system that uses the OSPF routing protocol. OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement the NetRanger system on network devices using OSPF.

8. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4164. NetRanger discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* NetRanger Anticipation Claim 4.

4165. To the extent this limitation is not disclosed by NetRanger, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

9. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4166. NetRanger discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* NetRanger Anticipation Claim 6.

4167. To the extent NetRanger does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, NetRanger in combination with RFC 2328 discloses this claim element. ’730 Patent claim 1; *see* NetRanger Anticipation Claims 1, 6; NetRanger Combination Claim 1; Ex. A-8, claim 6.

4168. NetRanger discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* NetRanger Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328.

4169. To the extent this limitation is not expressly disclosed by NetRanger, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. NetRanger discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

4170. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify NetRanger to create a corrective policy using Dijkstra’s Self Stabilization Algorithm in NetRanger’s multi-threaded distributed networking management system, as disclosed by RFC 2328.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. *Claim 7*

a. *A computer network, comprising:*

4171. NetRanger discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* NetRanger Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4172. NetRanger discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* NetRanger Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

4173. NetRanger discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* NetRanger Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

4174. NetRanger discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* NetRanger Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

4175. NetRanger discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* NetRanger Anticipation Claim 7.

f. *is able to clone itself;*

4176. NetRanger discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* NetRanger Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

4177. NetRanger discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* NetRanger Anticipation Claim 7.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4178. NetRanger discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* NetRanger Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

4179. NetRanger discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* NetRanger Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

4180. NetRanger discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* NetRanger Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4181. NetRanger discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* NetRanger Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

4182. NetRanger discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* NetRanger Anticipation Claim 7.

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4183. NetRanger discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* NetRanger Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4184. NetRanger discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* NetRanger Anticipation Claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4185. NetRanger discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* NetRanger Anticipation Claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications protocol encrypts a payload of a data packet.*

4186. NetRanger discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* NetRanger Anticipation Claim 13.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

4187. NetRanger discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* NetRanger Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4188. NetRanger discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* NetRanger Anticipation Claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4189. NetRanger discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* NetRanger Anticipation Claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4190. NetRanger discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* NetRanger Anticipation Claim 19. To the extent NetRanger does not by itself

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

anticipate this claim element, NetRanger in combination with Kasteleijn discloses this claim element. '730 Patent claim 19; *see* NetRanger Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-8, claim 19.

**19. Claim 21**

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4191. NetRanger discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* NetRanger Anticipation Claim 21.

**20. Claim 22**

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4192. NetRanger discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* NetRanger Anticipation Claim 22.

**21. Claim 24**

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4193. NetRanger discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* NetRanger Anticipation Claim 24.

**22. Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

4194. NetRanger discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* NetRanger Anticipation Claim 26.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

4195. NetRanger discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* NetRanger Anticipation Claim 29.

24. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4196. NetRanger discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* NetRanger Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4197. NetRanger discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* NetRanger Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

4198. NetRanger discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* NetRanger Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

4199. NetRanger discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* NetRanger Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is able to clone itself;*

4200. NetRanger discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* NetRanger Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4201. NetRanger discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* NetRanger Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4202. NetRanger discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See* NetRanger Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4203. NetRanger discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* NetRanger Anticipation Claim 30.

i. *including predicting a failure of a network component based on a predictive algorithm;*

4204. NetRanger discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* NetRanger Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

4205. NetRanger discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* NetRanger Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

4206. NetRanger discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* NetRanger Anticipation Claim 30.

25. ***Claim 31***

4207. NetRanger discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* NetRanger Anticipation Claim 31.

26. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4208. NetRanger discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* NetRanger Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

4209. NetRanger discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* NetRanger Anticipation Claim 32.

27. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4210. NetRanger discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* NetRanger Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4211. NetRanger discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* NetRanger Anticipation Claim 33.

28. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4212. NetRanger discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* NetRanger Anticipation Claim 34.

29. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4213. NetRanger discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* NetRanger Anticipation Claim 36.

**P. Cisco Catalyst in combination with Kasteleijn, and/or Goldman, and/or de**

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**Rocha, and/or NetRanger**

1. ***Motivation to Combine Cisco Catalyst with Kasteleijn***

4214. To the extent the ACL or QoS software in Cisco IOS 12.0 on the Catalyst 5000 Switch Family fails to explicitly or implicitly disclose a “software agent”, it would have been obvious to in view of Kasteleijn. *See* Ex. A-2, limitation 1.1. It would have been obvious to one of ordinary skill in the art to implement QoS or ACL software as a software agent to obtain the benefits of the MCs in Kasteleijn, for example, to maintain state during a resource crash or migration, and to improve resource reusage and reduce resource consumption.

2. ***Motivation to combine Cisco Catalyst with Goldman***

4215. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that will not result in failure), where modeling includes predicting failure of a network component (e.g. faults, inconsistencies or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings QoS and ACL agents), as taught by Goldman. *See* Ex. A-3, limitation 1.7.

3. ***Motivation to combine Cisco Catalyst with de Rocha***

4216. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. if faults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new ACL or QoS settings), as taught by da Rocha. *See* Ex. A-6, limitation 1.7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****4. *Motivation to combine Cisco Catalyst with NetRanger***

4217. To the extent this limitation is not explicitly or implicitly disclosed by Cisco Catalyst 5000 Switch Family, it would have been obvious to combine with NetRanger. A person of ordinary skill in the art would be motivated (as discussed in the NetRanger manual) to use Cisco Catalyst 5000 Switch Family devices as packet filtering devices monitored by NetRanger sensors for intrusion detection. See, Ex. A-8, limitation 1.7.

**5. *Claim 1*****a. *A method of managing a computer network, comprising:***

4218. Catalyst discloses this claim element. See Catalyst Anticipation Claim 1.

**b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment***

4219. Catalyst discloses this claim element. See Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; see Catalyst Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-9, claim 1.

**c. *is able to communicate with other software agents in the computer network;***

4220. Catalyst discloses this claim element. See Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; see Catalyst Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-9, claim 1.

**d. *is capable of perceiving its own state***

4221. Catalyst discloses this claim element. See Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; see Catalyst Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-9, claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *and is able to clone itself,*

4222. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* Catalyst Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-9, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4223. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* Catalyst Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-9, claim 1.

g. *monitoring the computer network;*

4224. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn and/or NetRanger discloses this claim element. '730 Patent claim 1; *see* Catalyst Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-8, claim 1; Ex. A-9, claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4225. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Goldman and/or de Rocha and/or NetRanger discloses this claim element. '730 Patent claim 1; *see* Catalyst Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1; Ex. A-8, claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

4226. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Goldman and/or de Rocha discloses this claim element. '730 Patent claim 1; *see* Catalyst Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1; Ex. A-9, claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4227. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Goldman and/or de Rocha discloses this claim element. '730 Patent claim 1; *see* Catalyst Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1; Ex. A-9, claim 1.

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

4228. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Goldman and/or de Rocha discloses this claim element. '730 Patent claim 1; *see* Catalyst Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-6, claim 1; Ex. A-9, claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4229. Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 1.

**6. Claim 2**

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4230. Catalyst discloses claim 2, "The method of claim 2, wherein the assigned goal of the Catalyst discloses this claim element. *See* Catalyst Anticipation Claim 2.

**7. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4231. Catalyst discloses this element of claim 3, "The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. *See* Catalyst Anticipation Claim 3.

- b. *constructing a topological representation of the computer network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

4232. Catalyst discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Catalyst Anticipation Claim 3.

8. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4233. Catalyst discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Catalyst Anticipation Claim 4.

9. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4234. Catalyst discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Catalyst Anticipation Claim 6.

10. ***Claim 7***

- a. *A computer network, comprising:*

4235. Catalyst discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* Catalyst Anticipation Claim 7.

- b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4236. Catalyst discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Catalyst Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

4237. Catalyst discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See Catalyst Anticipation Claim 7.*

d. *is able to communicate with other software agents in the computer network*

4238. Catalyst discloses this element of claim 7, “s able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See Catalyst Anticipation Claim 7.*

e. *is capable of perceiving its own state; and*

4239. Catalyst discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See Catalyst Anticipation Claim 7.*

f. *is able to clone itself;*

4240. Catalyst discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See Catalyst Anticipation Claim 7.*

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

4241. Catalyst discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See Catalyst Anticipation Claim 7.*

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4242. Catalyst discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See Catalyst Anticipation Claim 7.*

i. *said modeler comprising a predictive algorithm to predict a failure*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*of a network component*

4243. Catalyst discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Catalyst Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the prediction;*

4244. Catalyst discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Catalyst Anticipation Claim 7.

k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4245. Catalyst discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Catalyst Anticipation Claim 7.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4246. Catalyst discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Catalyst Anticipation Claim 7.

11. ***Claim 10***

a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4247. Catalyst discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Catalyst Anticipation Claim 10.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4248. Catalyst discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Catalyst Anticipation Claim 11. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. ’730 Patent claim 11; *see* Catalyst Anticipation Claim 11; Ex. A-2, claim 11; Ex. A-9, claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4249. Catalyst discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Catalyst Anticipation Claim 12. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. ’730 Patent claim 12; *see* Catalyst Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-9, claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

4250. Catalyst discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Catalyst Anticipation Claim 13.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*for a particular network component.*

4251. Catalyst discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Catalyst Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4252. Catalyst discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Catalyst Anticipation Claim 17. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. ’730 Patent claim 17; *see* Catalyst Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-9, claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4253. Catalyst discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Catalyst Anticipation Claim 18. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. ’730 Patent claim 18; *see* Catalyst Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-9, claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4254. Catalyst discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Catalyst Anticipation Claim 19. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn and/or Cisco IOS discloses this claim

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

element. '730 Patent claim 19; *see* Catalyst Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-9, claim 19.

19. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4255. Catalyst discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Catalyst Anticipation Claim 21. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn discloses this claim element. ’730 Patent claim 21; *see* Catalyst Anticipation Claim 21; Ex. A-2, claim 21; Ex. A-9, claim 21.

20. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4256. Catalyst discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Catalyst Anticipation Claim 22. To the extent Catalyst does not by itself anticipate this claim element, Catalyst in combination with Kasteleijn and/or NetRanger discloses this claim element. ’730 Patent claim 22; *see* Catalyst Anticipation Claim 22; Ex. A-2, claim 22; Ex. A-8, claim 22; Ex. A-9, claim 22.

21. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4257. Catalyst discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Catalyst Anticipation Claim 24.

22. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

4258. Catalyst discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Catalyst Anticipation Claim 26.

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

4259. Catalyst discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Catalyst Anticipation Claim 29.

24. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4260. Catalyst discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Catalyst Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4261. Catalyst discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Catalyst Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

4262. Catalyst discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Catalyst Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

d. *is capable of perceiving its own state; and*

4263. Catalyst discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See Catalyst Anticipation Claim 30.*

e. *is able to clone itself;*

4264. NetRa Catalyst nger discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See Catalyst Anticipation Claim 30.*

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4265. Catalyst discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See Catalyst Anticipation Claim 30.*

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4266. Catalyst discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See Catalyst Anticipation Claim 30.*

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4267. Catalyst discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See Catalyst Anticipation Claim 30.*

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

4268. Catalyst discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See Catalyst Anticipation Claim 30.*

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4269. Catalyst discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See Catalyst Anticipation Claim 30.*

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

4270. Catalyst discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See Catalyst Anticipation Claim 30.*

**25. Claim 31**

4271. Catalyst discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See Catalyst Anticipation Claim 31.*

**26. Claim 32**

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4272. Catalyst discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See Catalyst Anticipation Claim 32.*

b. *constructing a topological representation of the computer network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

4273. Catalyst discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Catalyst Anticipation Claim 32.

**27. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4274. Catalyst discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Catalyst Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4275. Catalyst discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Catalyst Anticipation Claim 33.

**28. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4276. Catalyst discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Catalyst Anticipation Claim 34.

**29. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Algorithm.*

4277. Catalyst discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Catalyst Anticipation Claim 36.

**Q. Cisco Adjacency in combination with Kasteleijn, and/or Pandya, and/or de Rocha, and/or Hellerstein, and/or RFC 2328**

**1. *Motivation to Combine Cisco Adjacency with Kasteleijn***

4278. To the extent this limitation is not disclosed by [REDACTED] [REDACTED] it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.1. It would have been obvious to implement [REDACTED] as a software agent to obtain the benefits of the MCs in Kasteleijn, for example, to maintain state during a resource crash or migration, and to improve resource reusage and reduce resource consumption.

**2. *Motivation to combine [REDACTED] with Pandya***

4279. To the extent this limitation is not disclosed by [REDACTED] [REDACTED] it is rendered obvious in combination with Pandya. *See* Ex. A-4, limitation 1.1. It would have been obvious to implement [REDACTED] as agents in communication with a control unit as taught by Pandya, in order to, for example, permit the [REDACTED] [REDACTED] software to be managed by an external entity or entities.

**3. *Motivation to combine [REDACTED] with de Rocha***

4280. To the extent this limitation is not met, it would have been obvious in view of da Rocha. *See* Ex. A-6, limitation 1.7. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (e.g. one that determined by corrective actions applied to proactively correct network errors), where modeling includes predicting failure of a network component (e.g. faults or events caused by, for example a flurry of errors), determining an appropriate policy based on the prediction, and finally dynamically

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

modifying the assigned goal of the software agents by deploying the optimal policy (e.g. new settings for [REDACTED]), as taught by da Rocha.

4. ***Motivation to combine [REDACTED] with NetRanger***

4281. To the extent this limitation is not explicitly or implicitly disclosed by [REDACTED], this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine [REDACTED] and Hellerstein to implement these models.

5. ***Motivation to combine [REDACTED] with RFC 2328***

4282. [REDACTED] in combination with RFC 2328 renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. [REDACTED] discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of [REDACTED] and RFC 2328. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328's self-stabilizing algorithms in [REDACTED] the RFC 2328 techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and [REDACTED] are found to not include each element of the asserted claims of the '730 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or [REDACTED] to include these elements, rendering them obvious.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6. *Claim 1*

a. *A method of managing a computer network, comprising:*

4283. [REDACTED] discloses this claim element. See [REDACTED] Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

4284. [REDACTED] discloses this claim element. See [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. '730 Patent claim 1; see [REDACTED] Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1; Ex. A-10, claim 1.

c. *is able to communicate with other software agents in the computer network;*

4285. [REDACTED] discloses this claim element. See [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. '730 Patent claim 1; see [REDACTED] Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1; Ex. A-10, claim 1.

d. *is capable of perceiving its own state*

4286. [REDACTED] discloses this claim element. See [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. '730 Patent claim 1; see [REDACTED] Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1; Ex. A-10, claim 1.

e. *and is able to clone itself,*

4287. [REDACTED] discloses this claim element. See [REDACTED] Anticipation Claim 1. To the extent [REDACTED] not by itself anticipate this claim [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. '730 Patent claim 1; see [REDACTED] Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1; Ex. A-10, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4288. [REDACTED] discloses this claim element. See [REDACTED] Anticipation Claim 1. To the extent [REDACTED] not by itself anticipate this claim element, [REDACTED] in combination

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

with Kasteleijn and/or Pandya discloses this claim element. '730 Patent claim 1; *see* [REDACTED] Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-4, claim 1; Ex. A-10, claim 1.

g. *monitoring the computer network;*

4289. [REDACTED] discloses this claim element. *See* [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* [REDACTED] Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-10, claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4290. [REDACTED] discloses this claim element. *See* [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with de Rocha and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* [REDACTED] Anticipation Claim 1; Ex. A-6, claim 1; Ex. A-10, claim 1; Ex. A-20, claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

4291. [REDACTED] discloses this claim element. *See* [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with de Rocha and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* [REDACTED] Anticipation Claim 1; Ex. A-6, claim 1; Ex. A-10, claim 1; Ex. A-20, claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4292. [REDACTED] discloses this claim element. *See* [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with de Rocha and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* [REDACTED] Anticipation Claim 1; Ex. A-6, claim 1; Ex. A-10, claim 1; Ex. A-20, claim 1.

k. *dynamically modifying the assigned goal of the software agent by*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*replacing the assigned goal based on the optimal policy*

4293. [REDACTED] discloses this claim element. *See* [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] combination with de Rocha and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* [REDACTED] Anticipation Claim 1; Ex. A-6, claim 1; Ex. A-10, claim 1; Ex. A-20, claim 1.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4294. [REDACTED] this claim element. *See* [REDACTED] Anticipation Claim 1. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with de Rocha discloses this claim element. '730 Patent claim 1; *see* [REDACTED] Anticipation Claim 1; Ex. A-6, claim 1; Ex. A-10, claim 1.

7. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4295. [REDACTED] discloses claim 2, "The method of claim 2, wherein the assigned goal of the [REDACTED] discloses this claim element. *See* [REDACTED] Claim 2.

8. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4296. [REDACTED] discloses this element of claim 3, "The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. *See* [REDACTED] Anticipation Claim 3.

4297. Furthermore, it would have been obvious to combine [REDACTED] with the OSFP routing protocol (RFC 2328). Under NetFuel's interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. [REDACTED]

[REDACTED] OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

ordinary skill in the art to implement [REDACTED] system on network devices using OSPF. See also CSI-NF-00000014 at PDF p.2.

- b. *constructing a topological representation of the computer network from the information.*

4298. [REDACTED] discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. [REDACTED] Anticipation Claim 3.

4299. Furthermore, it would have been obvious to combine [REDACTED] with the OSFP routing protocol (RFC 2328). Under NetFuel’s interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. [REDACTED]

[REDACTED] routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement [REDACTED] system on network devices using OSPF. See also CSI-NF-00000014 at PDF p.2.

9. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4300. [REDACTED] discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. See [REDACTED] Anticipation Claim 4.

4301. To the extent this limitation is not disclosed by [REDACTED], it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

10. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*Dijkstra Self Stabilization Algorithm.*

4302. [REDACTED] discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* [REDACTED] Anticipation Claim 6.

4303. To the extent [REDACTED] does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, [REDACTED] in combination with RFC 2328 discloses this claim element. ’730 Patent claim 1; *see* [REDACTED] Anticipation Claims 1, 6; [REDACTED] Combination Claim 6; Ex. A-10, claim 6.

4304. [REDACTED] discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* [REDACTED] Anticipation Claims 1, 2, 6. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328.

4305. To the extent this limitation is not expressly disclosed by [REDACTED] it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. [REDACTED]

[REDACTED]. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

4306. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify [REDACTED] a corrective policy using Dijkstra’s Self Stabilization Algorithm in [REDACTED] multi-threaded distributed networking management system, as disclosed by RFC 2328.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. *Claim 7*

a. *A computer network, comprising:*

4307. [REDACTED] discloses the preamble of claim 7, “[a] computer network, comprising.”  
’730 Pat. at Cl. 7. *See* [REDACTED] Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4308. [REDACTED] discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* [REDACTED] Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

4309. [REDACTED] discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* [REDACTED] Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

4310. [REDACTED] discloses this element of claim 7, “s able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* [REDACTED] Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

4311. [REDACTED] discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* [REDACTED] Anticipation Claim 7.

f. *is able to clone itself;*

4312. [REDACTED] discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* [REDACTED] Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

4313. [REDACTED] discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. See [REDACTED] Anticipation Claim 7.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4314. [REDACTED] discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. See [REDACTED] Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

4315. [REDACTED] discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. See [REDACTED] Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

4316. [REDACTED] discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. See [REDACTED] Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4317. [REDACTED] discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. See [REDACTED] Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

4318. [REDACTED] this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* [REDACTED] Anticipation Claim 7.

12. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4319. [REDACTED] discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* [REDACTED] Anticipation Claim 10.

13. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4320. [REDACTED] discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* [REDACTED] Anticipation Claim 11. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] combination with Kasteleijn and/or Pandya discloses this claim element. ’730 Patent claim 11; *see* [REDACTED] Anticipation Claim 11; Ex. A-2, claim 11; Ex. A-4, claim 11; Ex. A-10, claim 11.

14. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4321. [REDACTED] claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. [REDACTED] Anticipation Claim 12. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

element. '730 Patent claim 12; *see* [REDACTED] Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-4, claim 12; Ex. A-10, claim 12.

15. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

4322. [REDACTED] discloses claim 13, "[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet." '730 Pat. at Cl. 13. *See* [REDACTED] Anticipation Claim 13.

16. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

4323. [REDACTED] discloses claim 16, "[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component." '730 Pat. at Cl. 16. *See* [REDACTED] Anticipation Claim 16. To the extent [REDACTED] not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. '730 Patent claim 16; *see* [REDACTED] Anticipation Claim 16; Ex. A-2, claim 16; Ex. A-4, claim 11; Ex. A-10, claim 16.

17. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4324. [REDACTED] discloses claim 17, "[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria." '730 Pat. at Cl. 17. *See* [REDACTED] Anticipation Claim 17. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. '730 Patent claim 17; *see* [REDACTED] Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-4, claim 17; Ex. A-10, claim 17.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

18. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4325. [REDACTED] discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. See [REDACTED] Anticipation Claim 18. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] combination with Kasteleijn and/or Pandya discloses this claim element. ’730 Patent claim 18; see [REDACTED] Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-4, claim 18; Ex. A-10, claim 18.

19. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4326. [REDACTED] discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. [REDACTED] Anticipation Claim 19. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. ’730 Patent claim 19; see [REDACTED] Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-4, claim 19; Ex. A-10, claim 19.

20. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4327. [REDACTED] claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. See [REDACTED] Anticipation Claim 21. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. ’730 Patent claim 21; see [REDACTED] Anticipation Claim 21; Ex. A-2, claim 21; Ex. A-4, claim 21; Ex. A-10, claim 21.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

21. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4328. [REDACTED] discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* [REDACTED] Anticipation Claim 22. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. ’730 Patent claim 22; *see* [REDACTED] Anticipation Claim 22; Ex. A-2, claim 22; Ex. A-4, claim 22; Ex. A-10, claim 22.

22. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4329. [REDACTED] discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. [REDACTED] Claim 24.

23. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

4330. [REDACTED] discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* [REDACTED] Anticipation Claim 26. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. ’730 Patent claim 26; *see* [REDACTED] Anticipation Claim 26; Ex. A-2, claim 26; Ex. A-4, claim 26; Ex. A-10, claim 26.

24. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined task without having to request further policy.*

4331. [REDACTED] discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* [REDACTED] Anticipation Claim 29. To the extent [REDACTED] does not by itself anticipate this claim element, [REDACTED] in combination with Kasteleijn and/or Pandya discloses this claim element. ’730 Patent claim 29; *see* [REDACTED] Anticipation Claim 29; Ex. A-2, claim 29; Ex. A-4, claim 29; Ex. A-10, claim 29.

25. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4332. [REDACTED] discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. [REDACTED] Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4333. [REDACTED] this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. [REDACTED] Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

4334. [REDACTED] this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* [REDACTED] Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

4335. [REDACTED] discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* [REDACTED] Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is able to clone itself;*

4336. [REDACTED] discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* [REDACTED] Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4337. [REDACTED] discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* [REDACTED] Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4338. [REDACTED] this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See* [REDACTED] Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4339. [REDACTED] discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* [REDACTED] Anticipation Claim 30.

i. *including predicting a failure of a network component based on a predictive algorithm;*

4340. [REDACTED] discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* [REDACTED] Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*based on the prediction; and*

4341. [REDACTED] discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. See [REDACTED] Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

4342. [REDACTED] discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. See [REDACTED] Anticipation Claim 30.

26. **Claim 31**

4343. [REDACTED] discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. See [REDACTED] Anticipation Claim 31.

27. **Claim 32**

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4344. [REDACTED] discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. See [REDACTED] Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

4345. [REDACTED] discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. [REDACTED] Anticipation Claim 32.

28. **Claim 33**

- a. *The machine-readable storage medium of claim 30, further*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4346. [REDACTED] discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* [REDACTED] Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4347. [REDACTED] discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* [REDACTED] Anticipation Claim 33.

29. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4348. [REDACTED] discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* [REDACTED] Anticipation Claim 34.

30. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4349. [REDACTED] discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* [REDACTED] Anticipation Claim 36.

**R. Cisco Receive rACL in combination with Kasteleijn and/or NetRanger and/or Goldman**

- 1. ***Motivation to Combine Cisco Receive rACL in combination with***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY*****Kasteleijn***

A person of ordinary skill in the art would understand that an agent capable of communicating with other agents in improving the efficiency of agents in a distributed network. Further, it would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn. Furthermore, it would have been obvious to implement rACL software as a software agent to obtain the benefits of the MCs in Kasteleijn, for example, to maintain state during a resource crash or migration, and to improve resource reusage and reduce resource consumption.

2. ***Motivation to Combine Cisco Receive rACL in combination with NetRanger***

A person of ordinary skill in the art would be motivated (as discussed in the NetRanger manual) to use Cisco devices with rACLs as packet filtering devices monitored by NetRanger sensors for intrusion detection. In the combined system, rACL rules and policies, which are designed to filter, block, or otherwise rate-limit traffic, are test policies. Each test policy has values that are set, for example, for permitting or denying traffic and are set by default or modified or configured by a user. The test policies indicate how the traffic in the network should look. For example, particular traffic should not be permitted at all on certain ports, or the flow rates for a particular packet types should be at or below a specific threshold.

3. ***Motivation to Combine Cisco Receive rACL in combination with Goldman***

Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. See Ex. A-2, limitations 1.7-1.10.

4. ***For motivation to combine with RFC 2328 and/or Afek: copy and paste the following; delete reference to RFC or Afek if only one of the two***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*references is covered in the invalidity chart.*

**5. Claim 1**

**a. A method of managing a computer network, comprising:**

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1. To the extent rACL fails to explicitly or implicitly disclose a “software agent”, it would have been obvious to in view of Kasteleijn. *See* Ex. A-2, limitation 1.1. assigning a goal to a software [agent], wherein the software agent has its own runtime environment

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1.

**b. is able to communicate with other software agents in the computer network;**

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 1.2.

**c. is capable of perceiving its own state**

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 1.3.

**d. and is able to clone itself,**

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 1.4.

**e. and wherein the goal is a programmatic expression of a predefined task for the software agent; and**

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 1.5.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

f. monitoring the computer network;

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1. To the extent this limitation is not explicitly or implicitly disclosed by Cisco Receive ACL, it would have been obvious to combine with NetRanger. A person of ordinary skill in the art would be motivated (as discussed in the NetRanger manual) to use Cisco devices having rACL capabilities as packet filtering devices monitored by NetRanger sensors for intrusion detection. *See* Ex. A-8, limitation 1.6

g. creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,

Cisco Receive rACL discloses this claim element. *See* Cisco Receive rACL Anticipation Claim 1. To the extent this limitation is not explicitly or implicitly disclosed by Cisco Receive ACL, it would have been obvious to combine with NetRanger. A person of ordinary skill in the art would be motivated (as discussed in the NetRanger manual) to use Cisco devices with rACLs as packet filtering devices monitored by NetRanger sensors for intrusion detection. *See*, Ex. A-8, limitation 1.7. In the combined system, rACL rules and policies, which are designed to filter, block, or otherwise rate-limit traffic, are test policies. Each test policy has values that are set, for example, for permitting or denying traffic and are set by default or modified or configured by a user. The test policies indicate how the traffic in the network should look. For example, particular traffic should not be permitted at all on certain ports, or the flow rates for a particular packet types should be at or below a specific threshold. Execution and enforcement of rACL test policies then models and reflects the actual behavior of the traffic flow on the computer network which can be monitored by NetRanger to determine and then instantiate a revised, optimal policy based on certain triggering events. For example, when a policy violation is detected by NetRanger, the Director determines the specifics of a particular event that is triggered to select a specific policy, which may include shunning the traffic by reconfiguring the rACLs to deny that traffic.

Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. See Ex. A-2, limitations 1.7-1.10.

h. including predicting a failure of a network component based on a prediction algorithm

Cisco Receive rACL discloses this claim element. See Cisco Receive rACL Anticipation Claim 1. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. See Ex. A-8, limitation 1.8. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. See Ex. A-2, limitations 1.7-1.10.

i. wherein said modeling comprises determining appropriate policy based on the prediction; and

Cisco Receive rACL discloses this claim element. See Cisco Receive rACL Anticipation Claim 1. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. See Ex. A-8, limitation 1.9. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. See Ex. A-2, limitations 1.7-1.10.

j. dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy

Cisco Receive rACL discloses this claim element. See Cisco Receive rACL Anticipation Claim 1. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. See Ex. A-8, limitation 1.10. As discussed above, the NetRanger system determines an appropriate action

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

for a policy violation which may include reconfiguring the Cisco Receive ACL's settings. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. See Ex. A-2, limitations 1.7-1.10.

k. wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.

Cisco Receive rACL discloses this claim element. See Cisco Receive rACL Anticipation Claim 1. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. See Ex. A-8, limitation 1.11. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. See Ex. A-8, limitation 1.11.

6. ***Claim 2***

a. The method of claim 1, wherein the assigned goal of the agent is expressed as a policy.

Cisco Receive rACL discloses claim 2, "The method of claim 2, wherein the assigned goal of the aCisco Receive rACL discloses this claim element. See Cisco Receive rACL Anticipation Claim 2.

7. ***Claim 3***

a. The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and

Cisco Receive rACL discloses this element of claim 3, "The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. See Cisco Receive rACL Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. constructing a topological representation of the computer network from the information.

Cisco Receive rACL discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Cisco Receive rACL Anticipation Claim 3.

**8. Claim 4**

- a. The method of claim 1, wherein the modeling uses a numerical method.

Cisco Receive rACL discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Cisco Receive rACL Anticipation Claim 4. Additionally this limitation is rendered obvious in view of NetRanger. *See* Ex. A-8, limitation 4.0.

**9. Claim 6**

- a. The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.

Cisco Receive rACL discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Cisco Receive rACL Anticipation Claim 6. Additionally this limitation is rendered obvious in view of NetRanger. *See* Ex. A-8, limitation 6.0.

**10. Claim 7**

- a. A computer network, comprising:

Cisco Receive rACL discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* Cisco Receive rACL Anticipation Claim 7.

- b. a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware

Cisco Receive rACL discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Cisco Receive rACL Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- c. wherein the software agent has its own runtime environment

Cisco Receive rACL discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element

- b. *See* Cisco Receive rACL Anticipation Claim 7.

- d. is able to communicate with other software agents in the computer network

Cisco Receive rACL discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Cisco Receive rACL Anticipation Claim 7.

- e. is capable of perceiving its own state; and

Cisco Receive rACL discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Cisco Receive rACL Anticipation Claim 7.

- f. is able to clone itself;

Cisco Receive rACL discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Cisco Receive rACL Anticipation Claim 7.

- g. an agent support mechanism embodied in hardware to provide support to the agent;

Cisco Receive rACL discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Cisco Receive rACL Anticipation Claim 7.

- h. a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network

Cisco Receive rACL discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Cisco Receive rACL Anticipation Claim 7.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- i. said modeler comprising a predictive algorithm to predict a failure of a network component

Cisco Receive rACL discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Cisco Receive rACL Anticipation Claim 7.

- j. wherein the modeler determines appropriate policy based on the prediction;

Cisco Receive rACL discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Cisco Receive rACL Anticipation Claim 7.

- k. a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;

Cisco Receive rACL discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Cisco Receive rACL Anticipation Claim 7.

- l. wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.

Cisco Receive rACL discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Cisco Receive rACL Anticipation Claim 7.

**11. Claim 10**

- a. The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.

Cisco Receive rACL discloses claim 10, “[t]he computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Cisco Receive rACL Anticipation Claim 10.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

12. ***Claim 11***

- a. The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.

Cisco Receive rACL discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Cisco Receive rACL Anticipation Claim 11. To the extent the rACL software fails to explicitly or implicitly disclose a “software agent” that is capable of communicating with other software agents via a communications mechanism, it would have been obvious to in view of Kasteleijn. *See* Ex. A-2 at limitation 11.

13. ***Claim 12***

- a. The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.

Cisco Receive rACL discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Cisco Receive rACL Anticipation Claim 12. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 12.

14. ***Claim 16***

- a. The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.

Cisco Receive rACL discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Cisco Receive rACL Anticipation Claim 16.

15. ***Claim 17***

- a. The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.

Cisco Receive rACL discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

criteria.” ’730 Pat. at Cl. 17. *See* Cisco Receive rACL Anticipation Claim 17. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 17.

**16. Claim 18**

a. The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.

Cisco Receive rACL discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Cisco Receive rACL Anticipation Claim 18. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 18.

**17. Claim 19**

a. The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.

Cisco Receive rACL discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Cisco Receive rACL Anticipation Claim 19. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 19.

**18. Claim 21**

a. The computer network of claim 7, further comprising a policy database to store the policy for the agent.

Cisco Receive rACL discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Cisco Receive rACL Anticipation Claim 21. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 19.

**19. Claim 22**

a. The computer network of claim 7, wherein the network control mechanism includes the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

graphical user interface.

Cisco Receive rACL discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Cisco Receive rACL Anticipation Claim 22. To the extent rACL fails to explicitly or implicitly disclose this limitation, it is rendered obvious in view of Kasteleijn. *See* Ex. A-2, limitation 22.

20. ***Claim 24***

a. The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.

Cisco Receive rACL discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Cisco Receive rACL Anticipation Claim 24.

21. ***Claim 26***

a. The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.

Cisco Receive rACL discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Cisco Receive rACL Anticipation Claim 26. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 26.0.

22. ***Claim 29***

a. The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.

Cisco Receive rACL discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Cisco Receive rACL Anticipation Claim 29. The combination of Cisco Receive ACL and NetRanger satisfies this limitation. *See* Ex. A-8, limitation 29.0.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

23. *Claim 30*

- a. A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising

Cisco Receive rACL discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Cisco Receive rACL Anticipation Claim 30.

- b. assigning a goal to a software agent; wherein the software agent has its own runtime environment;

Cisco Receive rACL discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Cisco Receive rACL Anticipation Claim 30.

- c. is able to communicate with other software agents in the computer network

Cisco Receive rACL discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Cisco Receive rACL Anticipation Claim 30.

- d. is capable of perceiving its own state; and

Cisco Receive rACL discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Cisco Receive rACL Anticipation Claim 30.

- e. is able to clone itself;

Cisco Receive rACL discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Cisco Receive rACL Anticipation Claim 30.

- f. and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task

Cisco Receive rACL discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

least for all the reasons explained above regarding claim 1, element l. *See* Cisco Receive rACL Anticipation Claim 30.

g. and wherein the goal is a programmatic expression of a predefined task for the software agent

Cisco Receive rACL discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Cisco Receive rACL Anticipation Claim 30.

h. creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network

Cisco Receive rACL discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Cisco Receive rACL Anticipation Claim 30.

i. including predicting a failure of a network component based on a predictive algorithm;

Cisco Receive rACL discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Cisco Receive rACL Anticipation Claim 30.

j. wherein said modeling comprises determining appropriate policy based on the prediction; and

Cisco Receive rACL discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Cisco Receive rACL Anticipation Claim 30.

k. dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.

Cisco Receive rACL discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

reasons explained above regarding claim 1, element k. *See* Cisco Receive rACL Anticipation Claim 30.

**24. Claim 31**

Cisco Receive rACL discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Cisco Receive rACL Anticipation Claim 31.

**25. Claim 32**

a. The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and

Cisco Receive rACL discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Cisco Receive rACL Anticipation Claim 32.

b. constructing a topological representation of the computer network from the information.

Cisco Receive rACL discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Cisco Receive rACL Anticipation Claim 32.

**26. Claim 33**

a. The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and

Cisco Receive rACL discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Cisco Receive rACL Anticipation Claim 33.

b. determining an appropriate modification to the assigned goal based on the monitoring and the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

desired operational characteristic.

Cisco Receive rACL discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Cisco Receive rACL Anticipation Claim 33.

27. ***Claim 34***

a. The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.

Cisco Receive rACL discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Cisco Receive rACL Anticipation Claim 34.

28. ***Claim 36***

a. The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.

Cisco Receive rACL discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Cisco Receive rACL Anticipation Claim 36.

**S. NetRanger in combination with Buchanan/Cantrill/Berry and/or RFC 2328/Afek**

1. ***Motivation to Combine Cisco NetRanger with Buchanan, Cantrill, and/or Berry***

4350. Cisco NetRanger in combination with Buchanan, Cantrill, and/or Berry renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco NetRanger discloses using mobile software agents to monitor and correct faults in a distributed network; Cantrill discloses monitoring thread parameters to improve distributed network efficiency; Buchanan discloses adaptive network load balancing in a distributed network with



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

mobile agents; and Berry discloses monitoring thread parameters in a network performance monitoring system. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco NetRanger and Buchanan, Cantrill, and/or Berry. For example, a person of ordinary skill in the art would have been motivated to implement Buchanan and/or Cantrill's structure for monitoring numerous operating parameters of a single core in the multicore system disclosed in Cisco NetRanger because the Buchanan, and/or Cantrill system enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial. As another example, a person of ordinary skill in the art would have been motivated to implement one or Cantrill or Berry's systems enabled more operating parameters to be monitored and permitted load balancing across threads for improved system speed and efficiency, which one of ordinary skill in the art would have found beneficial.. Additionally, to the extent that Buchanan, Cantrill, and/or Berry and Cisco NetRanger are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify Buchanan, Cantrill, and/or Berry and/or Cisco NetRanger to include these elements, rendering them obvious.

**2. *Motivation to Combine Cisco NetRanger with RFC 2328 and/or Afek***

4351. Cisco NetRanger in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Cisco NetRanger discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Cisco NetRanger and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Cisco NetRanger because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Cisco NetRanger are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Cisco NetRanger to include these elements, rendering them obvious.

**T. Conti in combination with Kasteleijn and/or Goldman and/or Hellerstein and/or RFC 2328 and/or Afek**

**1. *Motivation to Combine Conti with Kasteleijn***

4352. It would have been obvious to permit software agents to communicate amongst each others, as the MCs of Kasteleijn do, in order to permit code or functionality downloads from one agent to another, or generally exchange messages (such as informing other agents that a console is requesting a certain type of information), which improves the efficiency of agents in a distributed network. Also, it would have been obvious to have the agents clone themselves, for example, when the size of the network grows. The combination further permits agents to propagate throughout the network as taught by Kasteleijn in order to, for example, permit an increased flexibility in the size of the network that is monitored. Further, it would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn. Further, it would have been obvious at the time of invention to one of ordinary skill in the art to utilize the secure communications channel of Kasteleijn to effect.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****2. *Motivation to Combine Conti with Goldman***

4353. Conti discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman.

**3. *Motivation to Combine Conti with Hellerstein***

4354. Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine Conti and Hellerstein to implement these models.

**4. *Motivation to Combine Conti with RFC 2328 and/or Afek***

4355. Conti in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Conti discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Conti and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in Conti because the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Conti are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Conti to include these elements, rendering them obvious.

5. ***Claim 1***

a. *A method of managing a computer network, comprising:*

4356. Conti discloses this claim element. *See* Conti Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

4357. Conti discloses this claim element. *See* Conti Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

4358. Conti discloses this claim element. *See* Conti Anticipation Claim 1.

d. *is capable of perceiving its own state*

4359. Conti discloses this claim element. *See* Conti Anticipation Claim 1.

e. *and is able to clone itself,*

4360. Conti discloses this claim element. *See* Conti Anticipation Claim 1. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* Conti Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-17, claim 1.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4361. Conti discloses this claim element. *See* Conti Anticipation Claim 1. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* Conti Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-17, claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

g. *monitoring the computer network;*

4362. Conti discloses this claim element. *See* Conti Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4363. Conti discloses this claim element. *See* Conti Anticipation Claim 1. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Conti Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-17, claim 1; Ex. A-20, claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

4364. Conti discloses this claim element. *See* Conti Anticipation Claim 1. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Conti Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-17, claim 1; Ex. A-20, claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4365. Conti discloses this claim element. *See* Conti Anticipation Claim 1. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Conti Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-17, claim 1; Ex. A-20, claim 1.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

4366. Conti discloses this claim element. *See* Conti Anticipation Claim 1. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Conti Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-17, claim 1; Ex. A-20, claim 1.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4367. Conti discloses this claim element. *See* Conti Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

6. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4368. Conti discloses claim 2, “The method of claim 2, wherein the assigned goal of the Conti discloses this claim element. *See* Conti Anticipation Claim 2. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Kasteleijn discloses this claim element. ’730 Patent claim 1; *see* Conti Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-17, claim 1.

7. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4369. Conti discloses this element of claim 3, “The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* Conti Anticipation Claim 1.

- b. *constructing a topological representation of the computer network from the information.*

4370. Conti discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Conti Anticipation Claim 3.

4371. To the extent this limitation is not disclosed by Conti, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). CONTI discloses agents traversing a network and collecting information for reporting back to a the master agent. It would have been obvious at the time of invention to use path information or other information from the agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**8. **Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4372. Conti discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Conti Anticipation Claim 4.

4373. To the extent this limitation is not disclosed by Goldman, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

4374. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Conti’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at p. 200, Sec. 1.

9. **Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4375. Conti discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Conti Anticipation Claim 6.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4376. To the extent Conti does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Conti in combination with RFC 2328 and/or Afek discloses this claim element. '730 Patent claim 1; *see* Conti Anticipation Claims 1, 6; Conti Combination Claim 1; Ex. A-17, claim 6.

4377. Conti discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Conti Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra's Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

4378. To the extent this limitation is not expressly disclosed by Conti, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel's Infringement Contentions for '730 Patent. Conti discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

4379. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra's Self Stabilization Algorithm in Conti's distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at p. 200, Sec. 1.

4380. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Conti to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Conti's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

10. ***Claim 7***

a. *A computer network, comprising:*

4381. Conti discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* Conti Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4382. Conti discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See* Conti Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

4383. Conti discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See* Conti Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

4384. Conti discloses this element of claim 7, "is able to communicate with other software agents in the computer network," at least for all the reasons explained above regarding claim 1, element c. *See* Conti Anticipation Claim 7.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

e. *is capable of perceiving its own state; and*

4385. Conti discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Conti Anticipation Claim 7.

f. *is able to clone itself;*

4386. Conti discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Conti Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

4387. Conti discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Conti Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4388. Conti discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Conti Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

4389. Conti discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Conti Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the prediction;*

4390. Conti discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Conti Anticipation Claim 7.

k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*the optimal policy;*

4391. Conti discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Conti Anticipation Claim 7.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4392. Conti discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Conti Anticipation Claim 7.

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4393. Conti discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Conti Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4394. Conti discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Conti Anticipation Claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism comprises a secure communications protocol.*

4395. Conti discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Conti Anticipation Claim 12. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Kasteleijn discloses this claim element. ’730 Patent claim 12; *see* Conti Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-17, claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

4396. Conti discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Conti Anticipation Claim 13.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

4397. Conti discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Conti Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4398. Conti discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Conti Anticipation Claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*selected from the group comprising of spawn, kill and suspend.*

4399. Conti discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Conti Anticipation Claim 18. To the extent Conti does not by itself anticipate this claim element, Conti in combination with Kasteleijn discloses this claim element. ’730 Patent claim 18; *see* Conti Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-17, claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4400. Conti discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Conti Anticipation Claim 19.

19. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4401. Conti discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Conti Anticipation Claim 21.

20. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4402. Conti discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Conti Anticipation Claim 22.

21. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*a Dijkstra Self Stabilization Algorithm.*

4403. Conti discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Conti Anticipation Claim 24.

**22. Claim 26**

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

4404. Conti discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Conti Anticipation Claim 26.

**23. Claim 29**

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

4405. Conti discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Conti Anticipation Claim 29.

**24. Claim 30**

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4406. Conti discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Conti Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4407. Conti discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Conti Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

4408. Conti discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Conti Anticipation Claim 30.

d. *is capable of perceiving its own state; and*

4409. Conti discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Conti Anticipation Claim 30.

e. *is able to clone itself;*

4410. Conti discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Conti Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4411. Conti discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* Conti Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4412. Conti discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See* Conti Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4413. Conti discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Conti Anticipation Claim 30.

i. *including predicting a failure of a network component based on a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predictive algorithm;*

4414. Conti discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Conti Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4415. Conti discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Conti Anticipation Claim 30.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

4416. Conti discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Conti Anticipation Claim 30.

25. ***Claim 31***

4417. Conti discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Conti Anticipation Claim 31.

26. ***Claim 32***

a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4418. Conti discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Conti Anticipation Claim 32.

b. *constructing a topological representation of the computer network*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

4419. Conti discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Conti Anticipation Claim 32.

**27. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4420. Conti discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Conti Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4421. Conti discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Conti Anticipation Claim 33.

**28. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4422. Conti discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Conti Anticipation Claim 34.

**29. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Algorithm.*

4423. Conti discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Conti Anticipation Claim 36.

**U. Castelli in combination with Kasteleijn and/or Turek and/or Goldman and/or RFC 2328 and/or Afek**

**1. *Motivation to Combine Castelli with Kasteleijn***

4424. It would have been obvious to permit software agents to communicate amongst each others, as the MCs of Kasteleijn do, in order to permit code or functionality downloads from one agent to another, or generally exchange messages (such as informing other agents that a console is requesting a certain type of information), which improves the efficiency of agents in a distributed network. Also, it would have been obvious to have the agents clone themselves, for example, when the size of the network grows. The combination further permits agents to propagate throughout the network as taught by Kasteleijn in order to, for example, permit an increased flexibility in the size of the network that is monitored. Further, it would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn. Further, it would have been obvious at the time of invention to one of ordinary skill in the art to utilize the secure communications channel of Kasteleijn to effect.

**2. *Motivation to Combine Castelli with Turek***

4425. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Castelli's network elements or monitor system to request further policy from the policy engine or the ADMIN system when it cannot resolve a fault or satisfy a performance target.

**3. *Motivation to Combine Castelli with Goldman***

4426. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

to the agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman. Further, a person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer “modified” but considered “new.”

4. ***Motivation to Combine Castelli with RFC 2328 and/or Afek***

4427. Castelli in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Castelli discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra’s algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Castelli and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek’s self-stabilizing algorithms in Castelli because the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Castelli are found to not include each element of the asserted claims of the ’659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Castelli to include these elements, rendering them obvious.

5. ***Claim 1***

a. ***A method of managing a computer network, comprising:***

4428. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

4429. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

- c. *is able to communicate with other software agents in the computer network;*

4430. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* Castelli Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-12, claim 1.

- d. *is capable of perceiving its own state*

4431. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

- e. *and is able to clone itself,*

4432. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see* Castelli Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-12, claim 1.

- f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4433. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

- g. *monitoring the computer network;*

4434. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4435. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

- i. *including predicting a failure of a network component based on a prediction algorithm*

4436. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4437. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

4438. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4439. Castelli discloses this claim element. *See* Castelli Anticipation Claim 1. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Turek discloses this claim element. '730 Patent claim 1; *see* Castelli Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-12, claim 1.

6. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4440. Castelli discloses claim 2, "The method of claim 2, wherein the assigned goal of the aCastelli discloses this claim element. *See* Castelli Anticipation Claim 2.

7. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4441. Castelli discloses this element of claim 3, "The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. *See* Castelli Anticipation Claim 1.

4442. Further, it would have been obvious to combine Castelli with the OSFP routing protocol (RFC 2328). Under NetFuel's interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. Thus it would have been obvious for SRA agents to also monitor network resources used by OSFP, and to use OSPF routing protocol. OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement Castelli's system on network devices using OSPF.

- b. *constructing a topological representation of the computer network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

4443. Castelli discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* Castelli Anticipation Claim 3.

4444. Further, it would have been obvious to combine Castelli with the OSFP routing protocol (RFC 2328). Under NetFuel’s interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. Thus it would have been obvious for SRA agents to also monitor network resources used by OSFP, and to use OSPF routing protocol. OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement Castelli’s system on network devices using OSPF.

8. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4445. Castelli discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Castelli Anticipation Claim 4.

4446. Further, it would have been obvious to combine Castelli with the OSFP routing protocol. Under NetFuel’s apparent interpretation, monitoring the same type of network activity used by OSFP, and using OSPF, satisfies this limitation. Thus it would have been obvious for SRA agents to also monitor network resources used by OSFP, and to use OSPF routing protocol. OSFP routing protocol was the prevalent gateway routing protocol of the time and as such it would have been obvious at the time of invention to one of ordinary skill in the art to implement Castelli’s system on network devices using OSPF.

9. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Dijkstra Self Stabilization Algorithm.*

4447. Castelli discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Castelli Anticipation Claim 6.

4448. To the extent Castelli does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Castelli in combination with Afek discloses this claim element. ’730 Patent claim 1; *see* Castelli Anticipation Claims 1, 6; Castelli Combination Claim 1; Ex. A-12, claim 6.

4449. Castelli discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Castelli Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

4450. To the extent this limitation is not expressly disclosed by Castelli, it would have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Castelli’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at p. 200, Sec. 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

10. *Claim 7*

a. *A computer network, comprising:*

4451. Castelli discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* Castelli Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4452. Castelli discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* Castelli Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

4453. Castelli discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Castelli Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer network*

4454. Castelli discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Castelli Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

4455. Castelli discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Castelli Anticipation Claim 7.

f. *is able to clone itself;*

4456. Castelli discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Castelli Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*support to the agent;*

4457. Castelli discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Castelli Anticipation Claim 7.

- h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4458. Castelli discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Castelli Anticipation Claim 7.

- i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

4459. Castelli discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Castelli Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

4460. Castelli discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Castelli Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4461. Castelli discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Castelli Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

4462. Castelli discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Castelli Anticipation Claim 7.

11. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4463. Castelli discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Castelli Anticipation Claim 10.

12. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4464. Castelli discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Castelli Anticipation Claim 11. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Kasteleijn discloses this claim element. ’730 Patent claim 11; *see* Castelli Anticipation Claim 11; Ex. A-2, claim 11; Ex. A-12, claim 11.

13. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4465. Castelli discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Castelli Anticipation Claim 12. To the extent Castelli does not by itself anticipate this claim

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

element, Castelli in combination with Kasteleijn discloses this claim element. '730 Patent claim 12; *see* Castelli Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-12, claim 12.

14. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

4466. Castelli discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Castelli Anticipation Claim 13.

15. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

4467. Castelli discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Castelli Anticipation Claim 16.

16. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4468. Castelli discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Castelli Anticipation Claim 17. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Kasteleijn discloses this claim element. ’730 Patent claim 17; *see* Castelli Anticipation Claim 17; Ex. A-2, claim 17; Ex. A-12, claim 17.

17. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4469. Castelli discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Castelli Anticipation Claim 18. To the extent Castelli does not by itself anticipate this claim

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

element, Castelli in combination with Kasteleijn discloses this claim element. '730 Patent claim 18; *see* Castelli Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-12, claim 18.

18. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4470. Castelli discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Castelli Anticipation Claim 19. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Kasteleijn discloses this claim element. ’730 Patent claim 19; *see* Castelli Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-12, claim 19.

19. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4471. Castelli discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Castelli Anticipation Claim 21.

20. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4472. Castelli discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Castelli Anticipation Claim 22.

21. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4473. Castelli discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Castelli Anticipation Claim 24.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

22. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

4474. Castelli discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Castelli Anticipation Claim 26. To the extent Castelli does not by itself anticipate this claim element, Castelli in combination with Goldman discloses this claim element. ’730 Patent claim 26; *see* Castelli Anticipation Claim 26; Ex. A-3, claim 26; Ex. A-12, claim 26.

23. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

4475. Castelli discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* Castelli Anticipation Claim 29.

24. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4476. Castelli discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Castelli Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4477. Castelli discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Castelli Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

4478. Castelli discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Castelli Anticipation Claim 30.

d. *is capable of perceiving its own state; and*

4479. Castelli discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Castelli Anticipation Claim 30.

e. *is able to clone itself;*

4480. Castelli discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Castelli Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4481. Castelli discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* Castelli Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4482. Castelli discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element g. *See* Castelli Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4483. Castelli discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Castelli Anticipation Claim 30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- i. *including predicting a failure of a network component based on a predictive algorithm;*

4484. Castelli discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Castelli Anticipation Claim 30.

- j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4485. Castelli discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Castelli Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

4486. Castelli discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Castelli Anticipation Claim 30.

25. ***Claim 31***

4487. Castelli discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Castelli Anticipation Claim 31.

26. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4488. Castelli discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Castelli Anticipation Claim 32.

- b. *constructing a topological representation of the computer network*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*from the information.*

4489. Castelli discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Castelli Anticipation Claim 32.

**27. Claim 33**

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4490. Castelli discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Castelli Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4491. Castelli discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Castelli Anticipation Claim 33.

**28. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4492. Castelli discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Castelli Anticipation Claim 34.

**29. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Algorithm.*

4493. Castelli discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Castelli Anticipation Claim 36.

**V. Lindskog in combination with****1. *Motivation to Combine Lindskog with Kasteleijn***

4494. It would have been obvious to permit software agents to communicate amongst each others, as the MCs of Kasteleijn do, in order to permit code or functionality downloads from one agent to another, or generally exchange messages (such as informing other agents that a console is requesting a certain type of information), which improves the efficiency of agents in a distributed network. Also, it would have been obvious to have the agents clone themselves, for example, when the size of the network grows. The combination further permits agents to propagate throughout the network as taught by Kasteleijn in order to, for example, permit an increased flexibility in the size of the network that is monitored. Further, it would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn. Further, it would have been obvious at the time of invention to one of ordinary skill in the art to utilize the secure communications channel of Kasteleijn to effect.

**2. *Motivation to Combine Lindskog with Goldman***

4495. Lindskog discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (*e.g.* incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

assigned goal of the software agents by deploying the optimal policy (*e.g.* modified or new policies), as taught by Goldman. Further, a person of ordinary skill would understand that an updated policy that crosses an arbitrary threshold of amount of parameters/actions replaced is no longer “modified” but considered “new.”

**3. *Motivation to Combine Lindskog with Hellerstein***

4496. Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine Lindskog and Hellerstein to implement these models.

4497. It would have been obvious at the time of invention to one of ordinary skill in the art to enable Lindskog’s agents to request further policy from another agent or network element when it cannot resolve a fault or satisfy a performance target.

**4. *Motivation to Combine Lindskog with Turek***

4498. A person of ordinary skill in the art would understand that network management tools may benefit from a GUI.

**5. *Motivation to Combine Lindskog with RFC 2328 and/or Afek***

4499. Lindskog in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. Lindskog discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra’s algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of Lindskog and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek’s self-stabilizing algorithms in Lindskog because

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and Lindskog are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or Lindskog to include these elements, rendering them obvious.

6. ***Claim 1***

a. *A method of managing a computer network, comprising:*

4500. Lindskog discloses this claim element. *See Lindskog Anticipation Claim 1.*

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

4501. Lindskog discloses this claim element. *See Lindskog Anticipation Claim 1.* To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see Lindskog Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-18, claim 1.*

c. *is able to communicate with other software agents in the computer network;*

4502. Lindskog discloses this claim element. *See Lindskog Anticipation Claim 1.*

d. *is capable of perceiving its own state*

4503. Lindskog discloses this claim element. *See Lindskog Anticipation Claim 1.*

e. *and is able to clone itself,*

4504. Lindskog discloses this claim element. *See Lindskog Anticipation Claim 1.* To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Kasteleijn discloses this claim element. '730 Patent claim 1; *see Lindskog Anticipation Claim 1; Ex. A-2, claim 1; Ex. A-18, claim 1.*

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4505. Lindskog discloses this claim element. *See Lindskog Anticipation Claim 1.*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

g. *monitoring the computer network;*

4506. Lindskog discloses this claim element. *See* Lindskog Anticipation Claim 1.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4507. Lindskog discloses this claim element. *See* Lindskog Anticipation Claim 1. To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Lindskog Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-18, claim 1; Ex. A-20, claim 1.

i. *including predicting a failure of a network component based on a prediction algorithm*

4508. Lindskog discloses this claim element. *See* Lindskog Anticipation Claim 1. To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Lindskog Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-18, claim 1; Ex. A-20, claim 1.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4509. Lindskog discloses this claim element. *See* Lindskog Anticipation Claim 1. To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Lindskog Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-18, claim 1; Ex. A-20, claim 1.

k. *dynamically modifying the assigned goal of the software agent by replacing the assigned goal based on the optimal policy*

4510. Lindskog discloses this claim element. *See* Lindskog Anticipation Claim 1. To the extent Lindskog does not by itself anticipate this claim element, Lindskog in combination with Goldman and/or Hellerstein discloses this claim element. '730 Patent claim 1; *see* Lindskog Anticipation Claim 1; Ex. A-3, claim 1; Ex. A-18, claim 1; Ex. A-20, claim 1.

l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform the predefined task.*

4511. Linskog discloses this claim element. *See* Linskog Anticipation Claim 1. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Turek discloses this claim element. '730 Patent claim 1; *see* Linskog Anticipation Claim 1; Ex. A-1, claim 1; Ex. A-18, claim 1.

7. ***Claim 2***

- a. *The method of claim 2, wherein the assigned goal of the agent is expressed as a policy.*

4512. Linskog discloses claim 2, "The method of claim 2, wherein the assigned goal of the aLinskog discloses this claim element. *See* Linskog Anticipation Claim 2. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Kasteleijn discloses this claim element. '730 Patent claim 2; *see* Linskog Anticipation Claim 2; Ex. A-2, claim 2; Ex. A-18, claim 2.

8. ***Claim 3***

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4513. Linskog discloses this element of claim 3, "The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task." '730 Pat. at Cl. 3. *See* Linskog Anticipation Claim 1.

- b. *constructing a topological representation of the computer network from the information.*

4514. Linskog discloses this element of claim 3, "constructing a topological representation of the computer network from the information." '730 Pat. at Cl. 3. *See* Linskog Anticipation Claim 3.

4515. To the extent this limitation is not disclosed by Linskog, it would have been obvious to combine Linskog with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2). Linskog discloses using agents to monitor and correct faults in a distributed network. It would have been obvious at the time of invention to use path information or other information from the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSFP messages to update the OSPF routing table.

9. ***Claim 4***

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4516. Lindskog discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* Lindskog Anticipation Claim 4.

4517. To the extent this limitation is not disclosed by Goldman, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSFP v2) . It would have been obvious to monitor OSPF routing protocol state changes using agents, and trigger new policy generation based on OSPF state changes.

4518. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Lindskog’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems.*** The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. ***The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter.*** For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at p. 200, Sec. 1.

10. ***Claim 6***

- a. *The method of claim 4, wherein the numerical method comprises a*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY***Dijkstra Self Stabilization Algorithm.*

4519. Linskog discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* Linskog Anticipation Claim 6.

4520. To the extent Linskog does not by itself anticipate this claim element or render it obvious in combination with the level of ordinary skill in the art at the time, Linskog in combination with RFC 2328 and/or Afek discloses this claim element. ’730 Patent claim 1; *see* Linskog Anticipation Claims 1, 6; Linskog Combination Claim 1; Ex. A-18, claim 6.

4521. Linskog discloses improving efficiency and scalability in a distributed network with agents having underlying resources. *See* Linskog Anticipation Claims 1, 2. It would have been obvious at the time of invention to a person of ordinary skill in the art to create corrective policy utilizing Dijkstra’s Self Stabilization Algorithm in order to achieve improved network efficiency and stability, as disclosed by RFC 2328 and/or Afek.

4522. To the extent this limitation is not expressly disclosed by Linskog, it would have been obvious to a person of ordinary skill in the art at the time in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). It is my understanding that NetFuel contends that the use of Open Shortest Path First (OSPF) satisfies this limitation. *See* NetFuel’s Infringement Contentions for ’730 Patent. Linskog discloses using agents to detect and correct faults in a distributed network. It would have been obvious at the time to detect changes in the OSPF routing table, such as OSPF adjacencies, to trigger corrective action.

4523. It would also have been obvious to a person of ordinary skill in the art to utilize the well-known Dijkstra’s Self Stabilization Algorithm in Linskog’s distributed network in order to achieve improved network efficiency and/or stability, as disclosed by Afek:

***In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*guarantee the eventual entry into a globally legal state and its maintenance thereafter.* For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].

Afek at p. 200, Sec. 1.

4524. Accordingly, for at least the reasons described above, it is my opinion that it would have been obvious at the time of invention to a person of ordinary skill in the art to modify Linskog to create a corrective policy using Dijkstra's Self Stabilization Algorithm in Linskog's multi-threaded distributed networking management system, as disclosed by RFC 2328 and/or Afek.

11. ***Claim 7***

a. *A computer network, comprising:*

4525. Linskog discloses the preamble of claim 7, "[a] computer network, comprising." '730 Pat. at Cl. 7. *See* Linskog Anticipation Claim 7.

b. *a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware*

4526. Linskog discloses this element of claim 7, "a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware," at least for all the reasons explained above regarding claim 1, element f. *See* Linskog Anticipation Claim 7.

c. *wherein the software agent has its own runtime environment*

4527. Linskog discloses this element of claim 7, "wherein the software agent has its own runtime environment," at least for all the reasons explained above regarding claim 1, element b. *See* Linskog Anticipation Claim 7.

d. *is able to communicate with other software agents in the computer*



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*network*

4528. Lindskog discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See* Lindskog Anticipation Claim 7.

e. *is capable of perceiving its own state; and*

4529. Lindskog discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See* Lindskog Anticipation Claim 7.

f. *is able to clone itself;*

4530. Lindskog discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See* Lindskog Anticipation Claim 7.

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

4531. Lindskog discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See* Lindskog Anticipation Claim 7.

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4532. Lindskog discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See* Lindskog Anticipation Claim 7.

i. *said modeler comprising a predictive algorithm to predict a failure of a network component*

4533. Lindskog discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* Lindskog Anticipation Claim 7.

j. *wherein the modeler determines appropriate policy based on the*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*prediction;*

4534. Lindskog discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* Lindskog Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4535. Lindskog discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* Lindskog Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4536. Lindskog discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* Lindskog Anticipation Claim 7.

12. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4537. Lindskog discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* Lindskog Anticipation Claim 10.

13. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*communications with agents.*

4538. Linskog discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* Linskog Anticipation Claim 11.

14. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4539. Linskog discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* Linskog Anticipation Claim 12. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Kasteleijn discloses this claim element. ’730 Patent claim 12; *see* Linskog Anticipation Claim 12; Ex. A-2, claim 12; Ex. A-18, claim 12.

15. ***Claim 13***

- a. *The computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.*

4540. Linskog discloses claim 13, “[t]he computer network of claim 12, wherein the secure communications protocol encrypts a payload of a data packet.” ’730 Pat. at Cl. 13. *See* Linskog Anticipation Claim 13. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Kasteleijn discloses this claim element. ’730 Patent claim 13; *see* Linskog Anticipation Claim 13; Ex. A-2, claim 13; Ex. A-18, claim 13.

16. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

4541. Linskog discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* Linskog Anticipation Claim 16.

17. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*predetermined criteria.*

4542. Linskog discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* Linskog Anticipation Claim 17.

18. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4543. Linskog discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* Linskog Anticipation Claim 18. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Kasteleijn discloses this claim element. ’730 Patent claim 18; *see* Linskog Anticipation Claim 18; Ex. A-2, claim 18; Ex. A-18, claim 18.

19. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4544. Linskog discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* Linskog Anticipation Claim 19. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Kasteleijn discloses this claim element. ’730 Patent claim 19; *see* Linskog Anticipation Claim 19; Ex. A-2, claim 19; Ex. A-18, claim 19.

20. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4545. Linskog discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* Linskog Anticipation Claim 21.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

21. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control mechanism includes the graphical user interface.*

4546. Linskog discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* Linskog Anticipation Claim 22. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Turek discloses this claim element. ’730 Patent claim 22; *see* Linskog Anticipation Claim 22; Ex. A-1, claim 22; Ex. A-18, claim 22.

22. ***Claim 24***

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4547. Linskog discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* Linskog Anticipation Claim 24.

23. ***Claim 26***

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

4548. Linskog discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* Linskog Anticipation Claim 26. To the extent Linskog does not by itself anticipate this claim element, Linskog in combination with Goldman discloses this claim element. ’730 Patent claim 26; *see* Linskog Anticipation Claim 26; Ex. A-3, claim 26; Ex. A-18, claim 26.

24. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

4549. Linskog discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

without having to request further policy.” ’730 Pat. at Cl. 29. *See* Lindskog Anticipation Claim 29.

25. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising*

4550. Lindskog discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* Lindskog Anticipation Claim 30.

- b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4551. Lindskog discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* Lindskog Anticipation Claim 30.

- c. *is able to communicate with other software agents in the computer network*

4552. Lindskog discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* Lindskog Anticipation Claim 30.

- d. *is capable of perceiving its own state; and*

4553. Lindskog discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* Lindskog Anticipation Claim 30.

- e. *is able to clone itself;*

4554. Lindskog discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* Lindskog Anticipation Claim 30.

- f. *and comprises an autonomous agent operable to request further*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy when it lacks an ability to perform the predefined task*

4555. Linskog discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element l. *See* Linskog Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined task for the software agent*

4556. Linskog discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* Linskog Anticipation Claim 30.

h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4557. Linskog discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* Linskog Anticipation Claim 30.

i. *including predicting a failure of a network component based on a predictive algorithm;*

4558. Linskog discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* Linskog Anticipation Claim 30.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4559. Linskog discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* Linskog Anticipation Claim 30.

k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*policy.*

4560. Lindskog discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* Lindskog Anticipation Claim 30.

26. ***Claim 31***

4561. Lindskog discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* Lindskog Anticipation Claim 31.

27. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4562. Lindskog discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* Lindskog Anticipation Claim 32.

- b. *constructing a topological representation of the computer network from the information.*

4563. Lindskog discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* Lindskog Anticipation Claim 32.

28. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4564. Lindskog discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* Lindskog Anticipation Claim 33.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4565. Linskog discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* Linskog Anticipation Claim 33.

29. ***Claim 34***

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4566. Linskog discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* Linskog Anticipation Claim 34.

30. ***Claim 36***

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4567. Linskog discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* Linskog Anticipation Claim 36.

**W. NSMIA in combination with Kasteleijn and/or Hellerstein and/or Goldman and/or RFC 2328 and/or Afek**

1. ***Motivation to Combine NSMIA in combination with Kasteleijn***

4568. A person of ordinary skill in the art would understand that an agent capable of communicating with other agents in improving the efficiency of agents in a distributed network. Further, it would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn. Furthermore, it would have been obvious to implement rACL software as a software agent to obtain the benefits of the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

MCs in Kasteleijn, for example, to maintain state during a resource crash or migration, and to improve resource reuse and reduce resource consumption.

**2. *Motivation to Combine NSMIA in combination with Hellerstein***

4569. A person of ordinary skill in the art would be motivated to use the model based system with NSMIA at the time of the invention, since Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine NSMIA and Hellerstein to implement these models.

**3. *Motivation to Combine NSMIA in combination with Goldman***

4570. Furthermore, this limitation would have been obvious in view of Goldman. It would have been obvious at the time of invention to a person of ordinary skill in the art to create a test policy for rACL and model the behavior of the network based on the test policy, including predicting a failure of a network device (e.g. misconfiguration, inconsistent policies, or other failures), in order to determine an optimal policy and appropriate policy, and dynamically update the rACL policies after testing with the optimal policy. See Ex. A-2, limitations 1.7-1.10.

4571. NSMIA in combination with RFC 2328 and/or Afek renders these claims obvious. A person of ordinary skill in the art would have been motivated to combine these references, including because these references are concerned with the same subject matter, management of distributed networks using software-agent-based scalable solutions. NSMIA discloses using mobile software agents to monitor and correct faults in a distributed network; RFC 2328 discloses the Open Shortest Path First algorithm; and Afek discloses using Dijkstra's algorithm in network management. The common subject matter and similar approaches to monitoring operating parameters in a distributed network, which were conceived and developed in a similar time frame, would have motivated one having ordinary skill in the art to combine the teachings of NSMIA and RFC 2328 and/or Afek. For example, a person of ordinary skill in the art would have been motivated to implement RFC 2328 and/or Afek's self-stabilizing algorithms in NSMIA because

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

the RFC 2328 and/or Afek techniques would have provided greater stability and efficiency in management of a distributed network, which one of ordinary skill in the art would have found beneficial. Additionally, to the extent that RFC 2328 and/or Afek and NSMIA are found to not include each element of the asserted claims of the '659 Patent, these elements were well known to one of ordinary skill in the art, and it would have been obvious to modify RFC 2328 and/or Afek and/or NSMIA to include these elements, rendering them obvious.

4. ***Claim 1***

a. *A method of managing a computer network, comprising:*

4572. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1.

b. *assigning a goal to a software [agent], wherein the software agent has its own runtime environment*

4573. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1.

c. *is able to communicate with other software agents in the computer network;*

4574. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1.

d. *is capable of perceiving its own state*

4575. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1.

e. *and is able to clone itself,*

4576. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.4.

f. *and wherein the goal is a programmatic expression of a predefined task for the software agent; and*

4577. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 1.5.

g. *monitoring the computer network;*

4578. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network,*

4579. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.7. NSMIA discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (e.g. incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. modified or new policies), as taught by Goldman.

4580. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine NSMIA and Hellerstein to implement these models.

- i. *including predicting a failure of a network component based on a prediction algorithm*

4581. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.8. NSMIA discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (e.g. incorrect configuration, policy inconsistencies, or other failures),

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. modified or new policies), as taught by Goldman.

4582. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine NSMIA and Hellerstein to implement these models.

j. *wherein said modeling comprises determining appropriate policy based on the prediction; and*

4583. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.9. NSMIA discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (e.g. incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. modified or new policies), as taught by Goldman.

4584. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine NSMIA and Hellerstein to implement these models.

k. *dynamically modifying the assigned goal of the software agent by*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*replacing the assigned goal based on the optimal policy*

4585. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1. To the extent this limitation is not met, it would have been obvious in view of Goldman. *See* Ex. A-3, limitation 1.10. NSMIA discloses that its agents may perform any number of policy-driven tasks, and that its agents may scale and adapt intelligently. It would have been obvious at the time of invention to a person of ordinary skill in the art to create test a policy and model the behavior of the computer network based on the test policy to determine an optimal policy (that is, one that will not cause device failure when deployed to the agents), where modeling includes predicting failure of a network component (e.g. incorrect configuration, policy inconsistencies, or other failures), determining an appropriate policy based on the prediction, and finally dynamically modifying the assigned goal of the software agents by deploying the optimal policy (e.g. modified or new policies), as taught by Goldman.

4586. Additionally, this limitation would be obvious in view of Hellerstein (Ex. A-20.) Hellerstein discloses creating models of metric values to construct and enforce alarm policies that automatically adjust to changes in configuration, topology, and workload, as well as warning policies based on the probability of violating an alarm policy. It would have been obvious at the time of invention to combine NSMIA and Hellerstein to implement these models and dynamically update policies based on them.

1. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4587. NSMIA discloses this claim element. *See* NSMIA Anticipation Claim 1.

5. ***Claim 2***

- a. *The method of claim 1, wherein the assigned goal of the agent is expressed as a policy.*

4588. NSMIA discloses claim 2, “The method of claim 2, wherein the assigned goal of the aNSMIA discloses this claim element. *See* NSMIA Anticipation Claim 2. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 2.0.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**6. Claim 3**

- a. *The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task; and*

4589. NSMIA discloses this element of claim 3, “The method of claim 1, further comprising: obtaining information about a network component by the software agent in performing the predefined task.” ’730 Pat. at Cl. 3. *See* NSMIA Anticipation Claim 1.

- b. constructing a topological representation of the computer network from the information.

4590. NSMIA discloses this element of claim 3, “constructing a topological representation of the computer network from the information.” ’730 Pat. at Cl. 3. *See* NSMIA Anticipation Claim 3. To the extent this limitation is not disclosed by NSMIA, it would have been obvious in combination with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). NSMIA discloses local layer agents traversing a network and collecting information for reporting back to the manager lawyer agents. It would have been obvious at the time of invention to use path information or other information from the agents to enhance an OSPF database. As each agent traverses individual links within the network, it may collect information as to the endpoints and states of those links. This information may be used to substitute or supplement link state OSPF messages to update the OSPF routing table.

**7. Claim 4**

- a. *The method of claim 1, wherein the modeling uses a numerical method.*

4591. NSMIA discloses claim 4, “The method of claim 1, wherein the modeling uses a numerical method.” ’730 Pat. at Cl. 4. *See* NSMIA Anticipation Claim 4. To the extent this limitation is not disclosed by NSMIA, it would have been obvious to combine NSMIA with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). NSMIA discloses agents that monitor the network. It would have been obvious to also consider OSPF routing protocol state changes, and trigger new policy generation based on OSPF state changes.

4592. Furthermore, modeling using numerical models such as the Dijkstra self-stabilization algorithm were well known at the time of invention. It would have been obvious, and

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

merely a design choice, to create a policy adaptation algorithm in NSMIA that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek.<sup>19</sup>

**Afek**

“In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter. For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].” (Afek at p. 200, Sec. 1) (emphasis added)

**8. Claim 6**

- a. *The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4593. NSMIA discloses claim 6, “The method of claim 4, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.” ’730 Pat. at Cl. 6. *See* NSMIA Anticipation Claim 6. To the extent this limitation is not disclosed by NSMIA, it would have been obvious to combine NSMIA with RFC 2328 (<https://tools.ietf.org/html/rfc2328>, OSPF v2). NSMIA discloses agents that monitor the network. It would have been obvious to also consider OSPF routing protocol state changes, and trigger new policy generation based on OSPF state changes.

4594. Furthermore, modeling using numerical models such as the Dijkstra self-stabilization algorithm were well known at the time of invention. It would have been obvious, and

<sup>19</sup> Yehuda Afek, Shay Kutten, and Moti Yung, The Local Detection Paradigm and its Applications to Self-Stabilization, Theoretical Computer Science Vol. 186, issue pp. 199-229 (published Oct. 30, 1997).



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

merely a design choice, to create a policy adaptation algorithm in NSMIA that uses the Dijkstra self-stabilization algorithm, as disclosed by Afek.<sup>20</sup>

**Afek**

“In 1974 Dijkstra suggested the notion of self-stabilizing systems. The notion is particularly interesting because of the above phenomenon: a system can be placed in an illegal global state, while each process is individually in a legal state. The self-stabilizing property assures that such a system automatically moves into and stays in a globally legal state regardless of its initial condition. The implementation of the self-stabilization methodology places a set of procedures, one at each processor that govern and dictate the necessary state transitions to guarantee the eventual entry into a globally legal state and its maintenance thereafter. For example, if a self-stabilizing token ring is placed in a state in which three tokens are present in the ring then the self-stabilizing algorithm automatically move the system into a state such that, in any subsequent state exactly one token is present in the ring, which circulates in a certain legal pattern. Since the introduction of this paradigm by Dijkstra in 1974 many self-stabilizing algorithms were developed to stabilize different distributed systems [2, 17, 18, 21,22, 25,26,27,29,30].” (Afek at p. 200, Sec. 1) (emphasis added)

**9. Claim 7****a. A computer network, comprising:**

4595. NSMIA discloses the preamble of claim 7, “[a] computer network, comprising.” ’730 Pat. at Cl. 7. *See* NSMIA Anticipation Claim 7.

**b. a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware**

4596. NSMIA discloses this element of claim 7, “a software agent having an assigned goal which is a programmatic expression of a predefined task for the software agent embodied in hardware,” at least for all the reasons explained above regarding claim 1, element f. *See* NSMIA Anticipation Claim 7.

<sup>20</sup> Yehuda Afek, Shay Kutten, and Moti Yung, The Local Detection Paradigm and its Applications to Self-Stabilization, Theoretical Computer Science Vol. 186, issue pp. 199-229 (published Oct. 30, 1997).

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

c. *wherein the software agent has its own runtime environment*

4597. NSMIA discloses this element of claim 7, “wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See NSMIA Anticipation Claim 7.*

d. *is able to communicate with other software agents in the computer network*

4598. NSMIA discloses this element of claim 7, “is able to communicate with other software agents in the computer network,” at least for all the reasons explained above regarding claim 1, element c. *See NSMIA Anticipation Claim 7.*

e. *is capable of perceiving its own state; and*

4599. NSMIA discloses this element of claim 7, “is capable of perceiving its own state,” at least for all the reasons explained above regarding claim 1, element d. *See NSMIA Anticipation Claim 7.*

f. *is able to clone itself;*

4600. NSMIA discloses this element of claim 7, “is able to clone itself,” at least for all the reasons explained above regarding claim 1, element e. *See NSMIA Anticipation Claim 7.*

g. *an agent support mechanism embodied in hardware to provide support to the agent;*

4601. NSMIA discloses this element of claim 7, “an agent support mechanism embodied in hardware to provide support to the agent.” ’730 Pat. at Cl. 7. *See NSMIA Anticipation Claim 7.*

h. *a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network*

4602. NSMIA discloses this element of claim 7, “be a modeler embodied in hardware to create test policy and to model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network,” at least for all the reasons explained above regarding claim 1, element h. *See NSMIA Anticipation Claim 7.*

i. *said modeler comprising a predictive algorithm to predict a failure*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*of a network component*

NSMIA discloses this element of claim 7, “said modeler comprising a predictive algorithm to predict a failure of a network component,” at least for all the reasons explained above regarding claim 1, element i. *See* NSMIA Anticipation Claim 7.

- j. *wherein the modeler determines appropriate policy based on the prediction;*

NSMIA discloses this element of claim 7, “wherein the modeler determines appropriate policy based on the prediction,” at least for all the reasons explained above regarding claim 1, element j. *See* NSMIA Anticipation Claim 7.

- k. *a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy;*

4603. NSMIA discloses this element of claim 7, “a network control mechanism to dynamically modify the assigned goal of the software agent by replacing the assigned goal based on the optimal policy,” at least for all the reasons explained above regarding claim 1, element k. *See* NSMIA Anticipation Claim 7.

- l. *wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task.*

4604. NSMIA discloses this element of claim 7, “wherein the software agent comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task,” at least for all the reasons explained above regarding claim 1, element l. *See* NSMIA Anticipation Claim 7.

10. ***Claim 10***

- a. *The computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network.*

4605. NSMIA discloses claim 10, “[t] computer network of claim 7, wherein the software agent comprises a monitoring agent having the assigned goal to monitor an operational characteristic of the network,” at least for all the reasons explained above regarding claim 1, element g. *See* NSMIA Anticipation Claim 10.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

11. ***Claim 11***

- a. *The computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.*

4606. NSMIA discloses claim 11, “[t]he computer network of claim 7, wherein the network control mechanism comprises a communications mechanism to facilitate communications with agents.” ’730 Pat. at Cl. 11. *See* NSMIA Anticipation Claim 11.

12. ***Claim 12***

- a. *The computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.*

4607. NSMIA discloses claim 12, “[t]he computer network of claim 11, wherein the communications mechanism comprises a secure communications protocol.” ’730 Pat. at Cl. 12. *See* NSMIA Anticipation Claim 12.

13. ***Claim 16***

- a. *The computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.*

4608. NSMIA discloses claim 16, “[t]he computer network of claim 7, wherein the agent support mechanism comprises an agent runtime environment configured for a particular network component.” ’730 Pat. at Cl. 16. *See* NSMIA Anticipation Claim 16. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 12.0.

14. ***Claim 17***

- a. *The computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.*

4609. NSMIA discloses claim 17, “[t]he computer network of claim 16, wherein the agent runtime environment controls an operation of the software agent based on predetermined criteria.” ’730 Pat. at Cl. 17. *See* NSMIA Anticipation Claim 17. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 18.0.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

15. ***Claim 18***

- a. *The computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.*

4610. NSMIA discloses claim 18, “[t]he computer network of claim 17, wherein the operation is selected from the group comprising of spawn, kill and suspend.” ’730 Pat. at Cl. 18. *See* NSMIA Anticipation Claim 18. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 18.0.

16. ***Claim 19***

- a. *The computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.*

4611. NSMIA discloses claim 19, “[t]he computer network of claim 16, wherein the agent runtime environment maintains a registry of all agents in the runtime environment.” ’730 Pat. at Cl. 19. *See* NSMIA Anticipation Claim 19. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 19.0. It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

17. ***Claim 21***

- a. *The computer network of claim 7, further comprising a policy database to store the policy for the agent.*

4612. NSMIA discloses claim 21, “[t]he computer network of claim 7, further comprising a policy database to store the policy for the agent.” ’730 Pat. at Cl. 21. *See* NSMIA Anticipation Claim 21. To the extent this limitation is not disclosed by NSMIA, it is rendered obvious in combination with Kasteleijn. *See* Ex. A-2, limitation 21.0 It would have been obvious at the time of invention to create a local or distributed directory of each agent running on each runtime environment as taught by Kasteleijn.

18. ***Claim 22***

- a. *The computer network of claim 7, wherein the network control*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*mechanism includes the graphical user interface.*

4613. NSMIA discloses claim 22, “[t]he computer network of claim 7, wherein the network control mechanism includes the graphical user interface.” ’730 Pat. at Cl. 22. *See* NSMIA Anticipation Claim 22. To the extent this limitation is not met, it would have been obvious in view of Turek. *See* Ex. A-1 at limitation 22.0. A person of ordinary skill in the art would understand that network management tools may benefit from a GUI.

19. Claim 24

- a. *The computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm.*

4614. NSMIA discloses claim 24, “[t]he computer network of claim 7, wherein the modeler comprises a Dijkstra Self Stabilization Algorithm,” at least for all the reasons explained above regarding claim 6. *See* NSMIA Anticipation Claim 24.

20. Claim 26

- a. *The computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.*

4615. NSMIA discloses claim 26, “[t]he computer network of claim 7, wherein the further policy is selected from the group comprising new policy and modified policy.” ’730 Pat. at Cl. 26. *See* NSMIA Anticipation Claim 26.

21. ***Claim 29***

- a. *The computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.*

4616. NSMIA discloses claim 29, “[t]he computer network of claim 7, wherein the software agent comprises an intelligent software agent operable to perform the predetermined task without having to request further policy.” ’730 Pat. at Cl. 29. *See* NSMIA Anticipation Claim 29.

22. ***Claim 30***

- a. *A machine-readable storage medium that provides instructions which when executed by a processor causes the processor to*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*perform a method comprising*

4617. NSMIA discloses the preamble of claim 30, “[a] machine-readable storage medium that provides instructions which when executed by a processor causes the processor to perform a method comprising.” ’730 Pat. at Cl. 30. *See* NSMIA Anticipation Claim 30.

b. *assigning a goal to a software agent; wherein the software agent has its own runtime environment;*

4618. NSMIA discloses this element of claim 30, “assigning a goal to a software agent; wherein the software agent has its own runtime environment,” at least for all the reasons explained above regarding claim 1, element b. *See* NSMIA Anticipation Claim 30.

c. *is able to communicate with other software agents in the computer network*

4619. NSMIA discloses this element of claim 30, “is able to communicate with other software agents in the computer network” at least for all the reasons explained above regarding claim 1, element c. *See* NSMIA Anticipation Claim 30.

d. *is capable of perceiving its own state; and*

4620. NSMIA discloses this element of claim 30, “is capable of perceiving its own state” at least for all the reasons explained above regarding claim 1, element d. *See* NSMIA Anticipation Claim 30.

e. *is able to clone itself;*

4621. NSMIA discloses this element of claim 30, “is able to clone itself” at least for all the reasons explained above regarding claim 1, element e. *See* NSMIA Anticipation Claim 30.

f. *and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task*

4622. NSMIA discloses this element of claim 30, “and comprises an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” at least for all the reasons explained above regarding claim 1, element f. *See* NSMIA Anticipation Claim 30.

g. *and wherein the goal is a programmatic expression of a predefined*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*task for the software agent*

4623. NSMIA discloses this element of claim 30, “and wherein the goal is a programmatic expression of a predefined task for the software agent” at least for all the reasons explained above regarding claim 1, element f. *See* NSMIA Anticipation Claim 30.

- h. *creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network*

4624. NSMIA discloses this element of claim 30, “creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network” at least for all the reasons explained above regarding claim 1, element h. *See* NSMIA Anticipation Claim 30.

- i. *including predicting a failure of a network component based on a predictive algorithm;*

4625. NSMIA discloses this element of claim 30, “including predicting a failure of a network component based on a predictive algorithm” at least for all the reasons explained above regarding claim 1, element i. *See* NSMIA Anticipation Claim 30.

- j. wherein said modeling comprises determining appropriate policy based on the prediction;  
and

4626. NSMIA discloses this element of claim 30, “wherein said modeling comprises determining appropriate policy based on the prediction” at least for all the reasons explained above regarding claim 1, element j. *See* NSMIA Anticipation Claim 30.

- k. *dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy.*

4627. NSMIA discloses this element of claim 30, “dynamically modifying the assigned goal of the software agent according to a desired operational characteristic of the computer network by replacing the assigned goal based on the optimal policy” at least for all the reasons explained above regarding claim 1, element k. *See* NSMIA Anticipation Claim 30.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

23. ***Claim 31***

4628. NSMIA discloses this element of claim 31 at least for all the reasons explained above regarding claim 1, element k. *See* NSMIA Anticipation Claim 31.

24. ***Claim 32***

- a. *The machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task; and*

4629. NSMIA discloses this element of claim 32, “[t]he machine-readable storage medium of claim 31, wherein the method further comprises obtaining information about a network component by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 3, element a. *See* NSMIA Anticipation Claim 32.

- b. constructing a topological representation of the computer network from the information.

4630. NSMIA discloses this element of claim 32, “constructing a topological representation of the computer network from the information” at least for all the reasons explained above regarding claim 3, element b. *See* NSMIA Anticipation Claim 32.

25. ***Claim 33***

- a. *The machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task; and*

4631. NSMIA discloses this element of claim 33, “[t]he machine-readable storage medium of claim 30, further comprising: monitoring an operational characteristic of the network by the software agent in performing the predefined task” at least for all the reasons explained above regarding claim 1, element g and claim 3, element a. *See* NSMIA Anticipation Claim 33.

- b. *determining an appropriate modification to the assigned goal based on the monitoring and the desired operational characteristic.*

4632. NSMIA discloses this element of claim 30, “determining an appropriate modification to the assigned goal based on the monitoring and the desired operational

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

characteristic” at least for all the reasons explained above regarding claim 1, elements h through k. *See* NSMIA Anticipation Claim 33.

**26. Claim 34**

- a. *The machine-readable storage medium of claim 33, wherein the determining uses a numerical method.*

4633. NSMIA discloses this element of claim 30, “The machine-readable storage medium of claim 33, wherein the determining uses a numerical method” at least for all the reasons explained above regarding claim 1, element 4. *See* NSMIA Anticipation Claim 34.

**27. Claim 36**

- a. *The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm.*

4634. NSMIA discloses this element of claim 30, “The machine-readable storage medium of claim 34, wherein the numerical method comprises a Dijkstra Self Stabilization Algorithm” at least for all the reasons explained above regarding claim 1, element 6. *See* NSMIA Anticipation Claim 36.

**XII. INVALIDITY UNDER OTHER GROUNDS**

**A. Subject Matter Eligibility - 35 U.S.C. § 101**

4635. It is my understanding that a patent claim is invalid if it is directed toward patent-ineligible subject matter. I further understand that determining whether a claim is directed towards a patent-ineligible abstract idea is a two-step process. It must be determined whether the claim is directed towards an abstract idea. If so, it must then be determined whether the claim contains an inventive concept sufficient to transform the claimed abstract idea into a patent-eligible application of that idea. My understanding is that implementing an abstract idea using well-known computer components or functions or taking an abstract idea and adding well-understood, routine, and conventional activities is not sufficient.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4636. It is my understanding that whether a claim is directed to eligible subject matter is a threshold legal issue for the Court. I nonetheless offer the following opinions regarding the subject-matter eligibility of the Asserted Claims.

4637. Having read the claims and specification of the '730 Patent, I expect to testify that the Asserted Claims of the '730 Patent are directed to the abstract idea of a generic prediction algorithm that "predicts" the failure of a network component, without providing any detail as to how it would accomplish the prediction or what constitutes a "failure." Other limitations similarly claim only the application of a vaguely-defined mathematical concept, such as a test policy to model the behavior of a network.

4638. It is also my opinion that the claim elements of the Asserted Claims of the '730 Patent do not add significantly more to these abstract ideas. Each element recites well understood, routine, and conventional computer elements. For example, at the time of the '730 Patent, agents, modelers, and prediction algorithms were well known and widely employed in software-defined networking.

4639. Having read the claims and specification of the '659 Patent, I expect to testify that the Asserted Claims of the '659 Patent are directed to the abstract idea of a generic corrective algorithm that performs a "corrective action to fix any detected abnormalities," without providing any detail as to how one would accomplish the corrective action nor to what would constitute a "detected abnormality." The claims are also directed to the abstract idea of delegating tasks to subordinates and having those subordinates ask their superior for help when they are unable to perform the task.

4640. It is also my opinion that the claim elements of the Asserted Claims of the '659 Patent do not add significantly more to these abstract ideas. Each element recites well understood, routine, and conventional computer elements. For example, at the time of the '659 Patent, agents, modelers, and methods of detecting problems in a network were well known and widely employed in software-defined networking.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**B. Inventorship - 35 U.S.C. § 102(f)**

4641. I understand that a correct inventorship is relevant to patent validity and enforceability. I understand that a patent must be filed in the name of all of the inventors of the claimed invention. I understand that nonjoinder is an inventorship error that occurs when the patent erroneously omits a person who is an inventor, and when such an error is not corrected or is uncorrectable, the patent is invalid. I also understand that inventorship errors may also render the patent unenforceable. I further understand the test for inventorship is whether a party had some conceptual role in at least an important or a necessary element, or important and necessary claim.

4642. In my opinion, the '730 and '659 Patents omit Mr. Yi-Hua Chang as an inventor. As discussed below, Mr. Chang contributed to the conception of claims of the '730 and '659 Patents. In forming my opinions on inventorship, I reviewed NetFuel's interrogatory responses regarding the alleged conception and reduction to practice of the alleged inventions of the '730 and '659 Patents. I also reviewed the deposition transcript and deposition exhibits of Mr. Yi-Hua Chang, as well as all of the documents and other evidence described below.

4643. It is my opinion that Mr. Chang contributed to the conception of at least claims 1, 7, and 30 of the '730 Patent, and therefore all the dependent claims that depend on claims 1, 7, and 30 of the '730 Patent. Mr. Chang testified at his deposition that in 2000-2001, Mr. Chang worked for NetFuel in connection with NetFuel's network management work. Chang Dep. Tr. 61:10-12; 26:25-27:12. Mr. Chang testified that he holds a Ph.D. and studied computer science in a post-doctoral program. Specifically, through 2001, Mr. Chang's testified that his primary role at NetFuel involved prediction and mitigation of future problems within a network, including "machine learning coding" intended to "try to find network system stability within network history." Chang Dep. Tr. at 22:5-23:15.

Q. You were trying to predict future problems?

A. Yeah. Future problem.

But that is ideally proposing, not -- just ideal of the network -

Q. Did that involve modeling a network?

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

A. Involving lot, lot, lot. But at that time --

Chang Dep. Tr. at 23:8–15.

Q. When -- so you mentioned that you were hired to work on neural -- using neural networks to predict future network problems?

A. Yeah. Ideally, the target is that.

MR. SEIGEL: Objection.

THE WITNESS: Target is try to do that, yes.

Chang Dep. Tr. at 27:6–12.

4644. In my opinion, a person of ordinary skill in the art at the time of invention of the subject matter of the '730 Patent would understand these issues to be significant evidence that the invention of the '730 Patent had neither yet been fully conceived or reduced to practice by the time Mr. Chang became involved, including with respect to use of a "prediction algorithm." Mr. Chang provided proposals for addressing these issues, including bringing ideas directly to Mr. Harlow for sign-off on engineering commits:

Q. And did you work on anything else while you were at NetFuel?

A. None. I think that is the target they want me working on.

Q. And what was Mr. Harlow's involvement in your work?

A. *I just either get something committed with him because he only the people I talk with on the company.* Other people, another people just involved business. Usually he not related to what I working on. Only James.

But he's still looking for mostly in some Google searches, find out what that he will -- he will bring me go to Berkeley, ask one professor, try to working with us. But I don't know finally is working or not.

Chang Dep. Tr. at 28:8–23.

4645. Mr. Chang further testified that he worked directly with the named inventor, Mr. James Harlow, while employed with NetFuel and was involved in at least one in-person person meeting with a Berkeley professor as part of an effort to use neural networks for predicting future network failures and detecting network problems. *See, e.g.,* Chang Dep. Tr. at 28:19-30:22.

*Q. Do you remember if you spoke with him about predicting future network problems?*

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*A. Predicting future network problem is a well-known subject of internet. Everybody knows that. Try working on this one.* Just which method you can use. This is a long, longtime subject. Lot of people working on the subject.

*Q. Did you talk to him about using special neural networks to detect network problem?*

*A. Yes.* This is special neural network is -- a European professor publish a book, not a secret. This means in one of the method of neural network. So not a secret. Not a patent. That is open, public domain.

The variety kind of neural network, you see lot of fails. So very few successful on neural network at that time, because lot like AI neural network here, very successful. At that time, neural network is not part subject.

Chang Dep. Tr. at 30:4-22.

4646. Mr. Chang also testified that his work contributed to other aspects of the claims of the '659 Patent, including claims regarding utilization of Dijkstra's algorithm. See, e.g., Chang Dep. Tr. at 48:1-50:9; *see also* Chang Dep. Tr., Ex. 103 at p. 34 (District Attorney's Report) ("Chang said he returned the CD containing Edsger Dijkstra notes to NetFuel in November of 2001.").

4647. 3 containing Edsger Dijkstra note

[Q] So Mr. Dijkstra was a professor?

A. Yeah. Professor at Texas. Very well-known people. You can search online. Very, very well-known people.

Q. Okay. And --

A. I think that time he already died. He's something like -- he's a system or something like a copy or CD-ROM mailed to me.

Q. He mailed you a CD-ROM?

A. A CD-ROM. A CD-ROM with a few paper involved in some question I ask. Because something I remember I do research on that.

Q. And that was something you were researching at the time that you were at NetFuel?

A. Yes.

Q. And so the Professor Dijkstra mailed you his CD-ROM with his paper?

A. Yeah. Well, the CD-ROM have lot, according his work and research. This is all public because professor. So that -- he mailed to me that CD-ROM.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Chang Dep. Tr. at 49:5-25.

4648. In my opinion, Mr. Chang, including through his work in coming up with ideas for a prediction algorithm to detect and predict network problems as well as in using Dijkstra's algorithm, contributed to the conception and reduction to practice of at least one material element of Claim 1 of the '730 Patent, including claim element 1.h: "creating test policy and modeling a behavior of the computer network based on the test policy to determine an optimal policy for the computer network, including predicting a failure of a network component based on a prediction algorithm." Particularly, it is my opinion that Mr. Chang's participation in meetings and discussions regarding the use of a prediction algorithm, his contribution to design and coding decisions and choices regarding the same, evidences his contribution to a material aspect of the claims of the '730 Patent. As a result of Mr. Chang's contributions, he is an inventor of every dependent claim to claims 1, 7, and 30.

4649. I also believe Mr. Chang contributed to the conception and reduction to practice of Claims 6, 24, and 36 of the '730 Patent, including claim elements 6.1, 24.1, and 36.1: "the numerical method comprises a Dijkstra Self Stabilization Algorithm." Similarly, it is my view that Mr. Chang contributed to the conception and reduction to practice of Claims 6, 12, and 186 of the '659 Patent, including claim elements 6.1, 12.1, and 18.1: "wherein creating the corrective policy comprises using Dijkstra's Self Stabilization Algorithmmm."

**C. Invalidity under 35 U.S.C. § 112**

**1. '730 Patent**

4650. In my opinion, the Asserted Claims of the '730 Patent are invalid for failing to comply with the written description and enablement requirements, because the specification does not support certain limitations of the Asserted Claims. Further, the '730 Patent specification does not support several limitations of the Asserted Claims as interpreted by NetFuel in its infringement contentions. Put another way, the products and systems in NetFuel's infringement contentions that are allegedly encompassed within the '730 Patent are not supported by the '730 specification.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

Finally, several of the limitations of the Asserted Claims are not sufficiently definite to permit one of ordinary skill in the art to make and use the claimed invention.

a. *Invalidity Under 35 U.S.C. § 112, ¶ 2—Written Description*

4651. I understand that under 35 U.S.C. § 112(a), claims of a patent are invalid if they fail to provide adequate written description of the claimed invention, including the process and manner of making and using it. This is so that the scope of the right to exclude set forth in the patent claims does not overreach the scope of the inventor's contribution to the field of art as described in the patent specification.

4652. In determining whether a patent meets this requirement, I understand one must account for the level of ordinary skill in the art at the time of the priority date of the patent. I further understand that there is an objective standard to determine compliance with the written description requirement. To satisfy the requirement, a patent's specification must convey with reasonable clarity to those of ordinary skill in the art that, as of the asserted priority date, the inventor(s) was/were in possession of the claimed invention.

4653. I understand that under the first paragraph of 35 U.S.C. § 112, a patent's specification must describe the claimed invention in such full, clear, concise, and exact terms as to enable one of ordinary skill in the art to make and use the full scope of the claimed invention. I also understand that in determining whether a patent meets this requirement, one takes into account the level of ordinary skill in the art as of the priority date of the patent. I further understand that a claim is invalid if it is not supported by an enabling disclosure.

4654. It is also my understanding that the fact that some experimentation may be required for one of ordinary skill in the art to practice the claimed invention does not necessarily mean the patent fails meet the enablement requirement. However, it is my understanding that factors that may be considered in determining whether undue experimentation would be required to make and use the claimed invention include: (1) the quantity of experimentation necessary; (2) the amount of direction or guidance disclosed in the patent; (3) the presence or absence of working examples in the patent; (4) the nature of the invention; (5) the state of the prior art; (6) the relative skill of



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

those in the art; (7) the predictability of the art; and (8) the breadth of the claims. It is further my understanding that patents are written for persons of skill in the field of the invention. Thus, a patent need not expressly state information that skilled persons would be likely to know or could obtain.

4655. I further understand that the full scope of the claims must be enabled, such that the scope of the claimed invention is not greater than the scope of the invention enabled by the specification. I also understand that elements of a claim may not be enabled when the inventor was not able to implement the elements at the time of filing. It is also my understanding that a patent can lack enablement of a given claim element by expressly teaching away from it.

- i. “to create a test policy and model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network”

4656. Independent claims 1, 7, and 30 of the '730 Patent require “creating test policy and modeling behavior of the computer network based on the test policy to determine an optimal policy of the computer network.” The '730 Patent's alleged invention allows a global modeler model a network in order to test a test policy using numerical modeling techniques so that an optimal policy may be determined prior to any actual implementation of the test/optimal policy:

This information is used by the global modeler module 18.4 to model network behavior. The module 18.3 allows test policy to be written and the effects of such test policy on the system 10 may be determined by using numerical modeling techniques in the module 18.4. In this way, policy may be tested before implementation in order to determine optimal policy.

'730 Pat. at 3:1-7.

4657. Further, as I noted above, I understand that the Court construed “optimal policy” as “a policy that is the most effective for a particular characteristic or set of characteristics.” In its infringement contentions, however, NetFuel appears to allege that default values/rules are test and optimal policies that satisfy this element of claim 1 of the '730 Patent. *See, e.g., '730 Infringement Chart at 43* (“For example, ***CoPP rules and policies, which are installed by default*** and designed to filter, block, or otherwise rate-limit traffic bound to the line card CPU or route processor, ***are test policies. Each test policy has values that are set by default or modified or configured by a***

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

*user.*”). In my view, the ’730 Patent specification does not support this interpretation of the claims. The patented system is always described in the specification as allowing a global modeler to model a system and to test the effects of a test policy so that an optimal policy may be determined **prior to implementation**, and there is no description of a system in the ’730 Patent that uses a default or user-configured rule as a test policy or that simply determines that an untested default or user-configured rule is an optimal policy. The patented system also does not describe how a default or user-configured rule could be found to be “a policy that is the most effective for a particular characteristic or set of characteristics,” as required by the Court’s construction of “optimal policy.” Moreover, execution of a default or user-configured rule is not modeling a network to determine an optimal policy, and there is no description of a system in the ’730 Patent that models the effects of a user-configured or default rule on the network by implementing that rule normally, outside of a testing or modeling environment.

4658. NetFuel also appears to contend that determining that a purported test policy is an optimal policy **after** implementation of said test policy by a switch/router satisfies this element of claim 1 of the ’730 Patent. *See, e.g.,* ’730 Infringement Chart at 48 (“**Execution and enforcement of CoPP’s test policies by CoPP then models and reflects the actual behavior of traffic flow on the computer network**, which can be monitored by a top talker / elephant trap algorithm that monitors punted traffic. **By sampling punted traffic, CoPP determines optimal policy** by predicting the presence of and identifying a bad actor—that is, an interface or sub-interface originating an excessive number of punt packets.”). But, in my view, the ’730 Patent specification also does not support this interpretation of the claims. The patented system is always described in the specification as allowing a global modeler to model a system and to test the effects of a test policy so that an optimal policy may be determined **prior to implementation**, and there is no description of a system in the ’730 Patent that simply determines that an untested default value is an optimal policy or that determines a purported test policy is an optimal policy after that test policy has already been implemented. Moreover, execution of a default setting is not modeling a network to determine an optimal policy, and there is no description of a system in the ’730 Patent

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

that models a network or tests a policy by actually executing the test policy in the network. The '730 Patent specification also does not explain how one would "create a test policy" or "model a behavior of the computer network based on the test policy" to arrive at any "policy," much less determining an "optimal policy of the computer network." For example, the specification only provides the above high-level description.

4659. Accordingly, if claims 1, 7, and 30 of the '730 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 30 are not supported by the written description in the '730 Patent specification or enabled.

- ii. "predicting a failure of a network component based on a prediction algorithm; wherein said modeling comprises determining appropriate policy based on the prediction"

4660. Claims 1, 7, and 30 include the limitation "predicting a failure of a network component based on a prediction algorithm; wherein said modeling comprises determining appropriate policy based on the prediction." '730 Pat. at cl. 1. The '730 Patent's alleged invention allows predicting a network failure using a prediction algorithm and determining an appropriate policy based on the same prediction:

At 284 the refinery creates one or more policies to try and fix the abnormality. It bases these policies upon its experience with similar problems. At 284 the refinery applies these corrective policies and at 290 the refinery observes the results of the corrective policies. The refinery learns by observing the new behavior of the system 10 and specifically the effect of the corrective policies on the system. In one embodiment of the invention, the refinery uses numerical methods to determine appropriate modifications to policy. The numerical methods may include the use of Kohonen Self Organizing maps and/or a Dijkstra Self Stabilization Algorithm. A Kohonen Self Organizing map is a neural network algorithm based on unsupervised learning. ***In another embodiment, the refinery, uses predictive algorithms to predict the failure of a network device and to determine appropriate corrective policy ahead of the failure.***

'730 patent at 23:44-59.

4661. In its infringement contentions, however, NetFuel appears to allege that sampling packets in a packet flow and applying a static top talker/elephant trap algorithm to identify bad actors satisfies this element of claim 1. *See, e.g., '730 Infringement Chart at 49.* In particular, NetFuel contends that:

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

The prediction algorithms used are the top talker / elephant trap algorithms. The purpose of predicting, identifying, and isolating a bad actor, along with further refinement as necessary by way of the dampening feature, is to determine an optimal policy that protects system resources, prevents a single interface or sub-interface from causing a failure by overwhelming the policer queue and rendering the device's control plane unavailable to other interfaces/subinterfaces (thereby reducing or curtailing service to others).

730 Infringement Chart at 49.

4662. But, in my view, the '730 Patent specification does not support this interpretation of the claims. The patented system is exclusively described in the specification as allowing uses predictive algorithms to predict the failure of a network device and to determine appropriate corrective policy ahead of the failure, and there is no description of a system in the '730 Patent that samples packets in a packet flow to identify the largest flows/flow sources according to a fixed rule. In fact, the word "packet[s]" only appears a single time in the specification. The specification also does not explain how to predict a network component failure using an elephant trap algorithm or how to model the network using an elephant trap algorithm. Moreover, the elephant trap algorithm is not a predictive algorithm, and there is no description in the '730 Patent of employing the elephant trap algorithm or any similar algorithm as a purported prediction algorithm. The specification also does not contain any written description of how an "appropriate policy" would be determined using an elephant trap algorithm or what that "appropriate policy" would be.

4663. Accordingly, if claims 1, 7, and 30 of the '730 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 30 are not supported by the written description in the '730 Patent specification or enabled.

iii. "is able to clone itself"

4664. The '730 Patent's alleged invention allows an agent to be able to clone itself. '730 Pat. at claim 1. In its infringement contentions, however, NetFuel appears to allege that an operating system restarting a process or maintaining a fail over or redundant processor or process satisfies this element of claim 1 of the '730 Patent. *See, e.g.,* '730 Infringement Chart at 38 ("For example, upon a component failure, **CoPP reinitializes itself**. See CSI-NF-00011567 at 10. CoPP also clones itself to create **redundant CoPP agents on the standby processors.**"). In my view, the

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

'730 Patent specification does not support this interpretation of the claims. The patented system is always described in the specification as requiring that an agent be capable of cloning itself, and there is no description of a system in the '730 Patent that uses a redundant or failover processor or that clones itself via restarting after a failure. Rather, the only mention of the word "clone" outside of the claims is a generic statement that an agent "is able to reproduce/clone itself." '730 Pat. at 7:36. Accordingly, if claims 1, 7, and 30 of the '730 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 30 are not supported by the written description in the '730 Patent specification or enabled.

- iv. "an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task"

Claims 1, 7, and 30 of the '730 Patent require "an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task." '730 Pat. at cl. 1. The '730 Patent's alleged invention allows an agent to request further policy when it lacks the policy to perform a predefined task. *See, e.g.*, '730 Patent at 11:54-58 ("In order for an agent to implement policy it may need to request services that are not currently running or loaded on the local ARE host. If the local ARE cannot support such an agent request, a request to the topology service is made to find a remote ARE that is either running or can run the required service." '730 Patent at 11:54-58.

In its infringement contentions, however, NetFuel appears to allege that sampling packets in a packet flow and applying a static top talker/elephant trap algorithm satisfies this element of claim 1. *See, e.g.*, '730 Infringement Chart at 49. In particular, NetFuel contends that:

In addition, in some instances CoPP lacks an ability to police bad actors across all interfaces on the Router where punt packets are from subscribers on a bundle member interface. In this situation, both the line card CoPP and route processor CoPP lack the ability to fully execute bad-actor policing and must request further policy both to determine whether a bundle member is active on a particular node and to receive instructions for when to "time out" the bad actor.

'730 Infringement Chart at 85. NetFuel also appears to contend that an alleged agent would lack the ability to perform a predefined task after "process restart":

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

In LPTS, PIFIB is an autonomous agent operable to request further policy when it lacks the ability to perform a predefined task. For example, upon process restart, PIFIB will request updates from the port arbitrator to update the PIFIB and reprogram the hardware and software policers.

'730 Infringement Chart at 167.

4665. But, in my view, the '730 Patent specification does not support such this interpretation of the claims. The specification does not contain any written description as to how the agent determines it “lacks an ability to perform the predefined task,” and there is no description of a system in the '730 Patent that determines that an alleged agent cannot perform a predefined task based on the agent having restarted or the agent purportedly requiring additional information regarding a “bundle member” on a separate node. Accordingly, if claim 1 of the '730 Patent is interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 30 are not supported by the written description in the '730 Patent specification or enabled.

v. “agent support mechanism”

4666. By way of further example, Claims 7 and 16 of the '730 Patent require an “agent support mechanism.” '730 Patent at Cl. 7. The '730 Patent's alleged invention requires “an agent support mechanism embodied in hardware to provide support to the agent.” '730 Patent at Cl. 7. Outside of the claims, the '730 Patent specification does not contain a single mention of the term “agent support mechanism” or even “agent support.” In its infringement contentions, however, NetFuel appears to allege that a “runtime environment” purportedly associated with CoPP or LPTS that “that provides resources to and has a form of state associated with processes running therein” constitutes an “agent support mechanism” and satisfies this element of claim 1 of the '730 Patent. *See, e.g.*, '730 Infringement Chart at 130 (“It has its own runtime environment that provides resources to and has a form of state associated with processes running therein.”). But, in my view, the '730 Patent specification also does not support this interpretation of the claims. The only mention of an “agent support mechanism” in the specification makes clear that it is “embodied in hardware,” and there is no description of a system in the '730 Patent that includes an “agent

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

runtime environment” embodied in hardware. Further, the specification does not explain how this “agent support mechanism” would be “embodied in hardware.”

4667. Accordingly, if claims 7 and 16 of the '730 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 7 and 16 are not supported by the written description in the '730 Patent specification or enabled.

- vi. “obtaining information about a network component by the software agent in performing the predefined task; and constructing a topological representation of the computer network from the information”

4668. Claims 3 and 33 of the '730 patent require “obtaining information about a network component by the software agent in performing the predefined task; and constructing a topological representation of the computer network from the information.” '730 Pat. at cl. 3. The '730 Patent’s alleged invention allows examination of the capabilities of a network device and its equipment and components and reporting of that information to a topology server for construction of a topological representation of the network:

The system 10 includes discovery agents which are used to examine and determine the capability of a network device (host) and the equipment or components of the device. Each discovery agent then reports this information to a topology server so that the system 10 can use the information to construct a topological representation of the network.

'730 Patent at 12:17-23

4669. Further, I understand that the Court construed “topological representation” as “[a] representation of the logical and/or physical arrangement of devices in a network.” In its infringement contentions, however, NetFuel appears to allege that maintaining information in a routing table satisfies this element of claim 1 of the '730 Patent.

Each Route Processor has its own view of the topology constructed by the shortest path first algorithm used by OSPF. The Route Processor is responsible for the routing table, while EEM possesses discovery capacity—i.e., obtaining information about a network component—through “event neighbor discovery”: <https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/eem/command/eem-cr-book/eem-cr-e2.html>, thereby constructing a topological representation of the network using link-state advertisement.

'730 Infringement Chart at 38.



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4670. In my view, the '730 Patent specification does not support this interpretation of the claims. The patented system is exclusively described in the specification as requiring examination of the capabilities of a network device and its equipment and components and reporting of that information to a topology server for construction of a topological representation of the network, and there is no description of a system in the '730 Patent that uses a routing table or the open shortest path first algorithm as a topological representation of a network. The '730 Patent specification also does not describe a routing table as being "[a] representation of the logical and/or physical arrangement of devices in a network," as required by the Court's construction. Accordingly, if claims 1, 7, and 30 of the '730 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 30 are not supported by the written description in the '730 Patent specification or enabled.

b. *Invalidity Under 35 U.S.C. § 112, ¶ 1—Enablement*

4671. I understand that under the first paragraph of 35 U.S.C. § 112, a patent's specification must describe the claimed invention in such full, clear, concise, and exact terms as to enable one of ordinary skill in the art to make and use the full scope of the claimed invention. I also understand that in determining whether a patent meets this requirement, one takes into account the level of ordinary skill in the art as of the priority date of the patent. I further understand that a claim is invalid if it is not supported by an enabling disclosure.

4672. It is also my understanding that the fact that some experimentation may be required for one of ordinary skill in the art to practice the claimed invention does not necessarily mean the patent fails meet the enablement requirement. However, it is my understanding that factors that may be considered in determining whether undue experimentation would be required to make and use the claimed invention include: (1) the quantity of experimentation necessary; (2) the amount of direction or guidance disclosed in the patent; (3) the presence or absence of working examples in the patent; (4) the nature of the invention; (5) the state of the prior art; (6) the relative skill of those in the art; (7) the predictability of the art; and (8) the breadth of the claims. It is further my understanding that patents are written for persons of skill in the field of the invention. Thus, a



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

patent need not expressly state information that skilled persons would be likely to know or could obtain.

4673. I further understand that the full scope of the claims must be enabled, such that the scope of the claimed invention is not greater than the scope of the invention enabled by the specification. I also understand that elements of a claim may not be enabled when the inventor was not able to implement the elements at the time of filing. It is also my understanding that a patent can lack enablement of a given claim element by expressly teaching away from it.

- i. “to create a test policy and model a behavior of the computer network based on the test policy thereby to determine an optimal policy for the computer network”

4674. As explained above, the patent does not contain a written description of an “optimal policy” nor how to determine an “optimal policy” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Similarly, because the patent contains no description of such a “optimal policy for the computer network” or the method for determining this “optimal policy,” it is also my view that the ’730 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the term “optimal policy” or “test policy” or “model a behavior.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the “optimal policy” or “test policy” or “model a behavior” lie.

- ii. “predicting a failure of a network component based on a prediction algorithm; wherein said modeling comprises determining appropriate policy based on the prediction”

4675. As described above, the patent does not contain a written description for “predicting a failure of a network component based on a prediction algorithm; wherein said modeling comprises determining appropriate policy based on the prediction” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such a “prediction algorithm” or the method for using this “prediction algorithm,” it is also my view that the ’730 Patent is not enabled. In particular, there

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the terms “prediction algorithm,” “appropriate policy,” or “modeling.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the “prediction algorithm,” “appropriate policy,” or “modeling” lie.

iii. “is able to clone itself”

4676. As described above, the patent does not contain a written description for “is able to clone itself” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such an “clone itself” or how an agent would “clone itself,” it is also my view that the ’730 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the term “clone itself.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the limitation lie.

iv. “an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task”

4677. As described above, the patent does not contain a written description for “an autonomous agent operable to request further policy when it lacks an ability to perform the predefined task” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such an “autonomous agent” or how it determines that it “lacks an ability to perform the predefined task,” it is also my view that the ’730 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the terms “autonomous agent” and “lacks an ability to perform the predefined task.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the limitation lie.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

## v. “agent support mechanism”

4678. As described above, the patent does not contain a written description for “agent support mechanism” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such a “prediction algorithm” or the method for using this “agent support mechanism,” it is also my view that the ’730 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the term “agent support mechanism.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because the claims do not set any clear boundaries for this limitation.

## vi. “obtaining information about a network component by the software agent in performing the predefined task; and constructing a topological representation of the computer network from the information”

4679. As described above, the patent does not contain a written description for “obtaining information about a network component by the software agent in performing the predefined task; and constructing a topological representation of the computer network from the information” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such a “topological representation” or “information about a network component,” it is also my view that the ’730 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the terms “topological representation” or “information about a network component.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the claim element terms lie.

2. ***’659 Patent***

4680. In my opinion, the Asserted Claims of the ’730 Patent are invalid for failing to comply with the written description and enablement requirements, because the specification does not support certain limitations of the Asserted Claims. Further, the ’730 Patent specification does

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

not support several limitations of the Asserted Claims as interpreted by NetFuel in its infringement contentions. Put another way, the products and systems in NetFuel's infringement contentions that are allegedly encompassed within the '730 Patent are not supported by the '730 specification. Finally, several of the limitations of the Asserted Claims are not sufficiently definite to permit one of ordinary skill in the art to make and use the claimed invention.

a. *Invalidity Under 35 U.S.C. § 112, ¶ 2—Written Description*

4681. I understand that under 35 U.S.C. § 112(a), claims of a patent are invalid if they fail to provide adequate written description of the claimed invention, including the process and manner of making and using it. This is so that the scope of the right to exclude set forth in the patent claims does not overreach the scope of the inventor's contribution to the field of art as described in the patent specification.

4682. In determining whether a patent meets this requirement, I understand one must account for the level of ordinary skill in the art at the time of the priority date of the patent. I further understand that there is an objective standard to determine compliance with the written description requirement. To satisfy the requirement, a patent's specification must convey with reasonable clarity to those of ordinary skill in the art that, as of the asserted priority date, the inventor(s) was/were in possession of the claimed invention.

i. “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread”

4683. Claims 1, 7, and 13 of the '659 Patent require “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread.” The '659 Patent's alleged invention allows monitoring of a per-thread utilization for threads running within an agent runtime environment. *See, e.g.*, '659 Patent at 11:27-30 (“As discussed above each ARE has the ability to provide feedback on the per-thread utilization of each thread running within it. This information is used by the CCE-20 to, for example, help determine load balancing on the ARE's.”). The specification only contains one mention of the term “per-thread utilization” outside of the claims. *See* '659 Patent at 11:27-30.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4684. As I noted above, I understand that the Court construed “thread” as “the execution of a section of code independently within a software program.” In its infringement contentions, however, NetFuel appears to allege that publishing an event when either CPU utilization or memory use for an Cisco IOS software process crosses a threshold satisfies this element of claim 1 of the '659 Patent. *See, e.g.*, '659 Infringement Chart at 14. But, in my view, the '659 Patent specification does not support this interpretation of the claims. The patented system is exclusively described in the specification as allowing monitoring operational parameters, including a per-thread utilization of each thread running within an agent runtime environment, and there is no description of a system in the '659 Patent that tracks CPU utilization or memory use of a process. In fact, “CPU utilization” and “memory use” are never mentioned in the '659 Patent specification.

4685. Accordingly, if claims 1, 7, and 13 of the '659 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 13 are not supported by the written description in the '659 Patent specification or enabled.

- ii. “detecting if there is an abnormality in the monitored operational parameters”

4686. Claims 1, 7, and 13 of the '659 Patent require “detecting if there is an abnormality in the monitored operational parameters.” The '659 Patent’s alleged invention allows monitoring of operational parameters and identifying an “abnormality” using a policy refinery. *See, e.g.*, '659 Patent at FIG. 18 (showing step 282 “refinery notices abnormality”); *see also id.* at 23:59-61 (“At 282 the refinery detects abnormalities in the system. At 284 the refinery creates one or more policies to try and fix 60 the abnormality.”). Further, the '659 Patent allows for the “monitored operational parameters” to be “relating to the each thread.” '659 Patent at cl. 1.

4687. In its infringement contentions, however, NetFuel appears to allege that a “high error rate related to data traffic” satisfies this element of claim 1 of the '659 Patent. *See, e.g.*, '659 Infringement Chart at 18 (“EEM can detect a high error rate related to data traffic monitored by CoPP”); '659 Infringement Chart at 19 (“EEM may be configured to detect abnormally high levels

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

of traffic passing through the control plane to the CPU"). In my view, the '659 Patent specification does not support this interpretation of the claims.

4688. NetFuel also contends that

CoPP and EEM detect abnormalities in the monitored operational parameters of CoPP, for instance monitoring and parsing statistics collected by CoPP such as punting, dropping, or rate limiting of packets, which can be flagged using EEM event detectors including SNMP and syslog.

'659 Infringement Chart at 19.

4689. NetFuel also apparently contends that. *See, e.g.*, '659 Infringement Chart at 20 ("For instance, in IOS XR devices, Cisco developed an EEM-based TCL script to monitor network behavior by calculating drop rates for packets dropped by LPTS."). These infringement theories apparently advanced by NetFuel require that "the monitored operational parameters [relating to the each thread]" include observing error rates and statistics regarding traffic and punting, dropping, or rate limiting of packets. But, in my view, the '659 Patent specification also does not support this interpretation of the claims. The patented system is exclusively described in the specification as allowing monitoring operational parameters relating to each thread running within an agent runtime environment, and there is no description of a system in the '659 Patent that tracks error rates and statistics regarding traffic and punting, dropping, or rate limiting of packets. The specification also includes no written description of what would be an "abnormality" in error rates and statistics regarding traffic and punting, dropping, or rate limiting of packets. In fact, the word "abnormality" appears only twice in the specification, outside of the claims, and neither mention sheds any light as to what might constitute an "abnormality."

4690. Accordingly, if claims 1, 7, and 13 of the '659 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 13 are not supported by the written description in the '659 Patent specification or enabled.

iii. "performing a corrective action to fix any detected abnormalities"

4691. Claims 1, 7, and 13 of the '659 Patent require "performing a corrective action to fix any detected abnormalities." The '659 Patent's alleged invention allows for a feedback loop in

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

which an agent detects an abnormality, reports the abnormality upwards to a global modeler, which sends a “corrective policy” down to an agent, which then performs a “corrective action” to address the detected abnormality. *See, e.g.*, ’659 Patent at FIG. 17A (showing step 250 “Tell Policy Agent to Perform Corrective Action”); *see also id.* at 11:41-46 (“Agents use a two stage feedback mechanism, which links the local modelers to the agents to provide the alert notification feedback loop required to monitor device/application evaluation and administer corrective actions within the network for a given network state.”). In its infringement contentions, however, NetFuel appears to allege that the purported global modeler both detects and provides policy to address problems. For example, NetFuel contends that “For example, when EEM detects a previously unknown threat, it will modify the threshold/policy for the network. It uses events to trigger a corrective policy” and that “EEM is a global modeler configured to listen to events from a system comprising a plurality of runtime environments, including the first runtime environment in which the agent (CoPP or LPTS) operates.” ’659 Infringement Chart at 20, 35. But, in my view, the ’659 Patent specification does not support this interpretation of the claims. The patented system is exclusively described in the specification as allowing a feedback loop in which the agent detects and addresses problems in accordance with the “corrective policy” received from the global modeler, and there is no description of a system in the ’659 Patent in which the global modeler performs the detection/monitoring elements of claims 1, 7, and 13.

4692. Further, NetFuel appears to content in its infringement contentions that a “corrective action” can be blocking or rate limiting traffic:

EEM and CoPP can thus perform a corrective action by modifying CoPP policies to block or rate limit traffic so as to more effectively police unwanted traffic, reduce unintended packet drops, or more appropriately rate limit traffic destined to the control plane.

’659 Infringement Chart at 24. In my view, the ’659 Patent specification also does not support this interpretation of the claims. The patented system is exclusively described in the specification as allowing a feedback loop in which the agent detects and addresses problems in accordance with the “corrective policy” received from the global modeler, and there is no description of a system

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

in the '659 Patent in which the global modeler performs the “corrective action” in addition to providing the “corrective policy” of claims 1, 7, and 13. The specification also does not any system in which blocking or rate limiting traffic are “corrective actions” or how blocking or rate limiting traffic would be employed in order to “fix any detected abnormalities.” *See, e.g.*, '659 Patent at Claim 1.

4693. Accordingly, if claims 1, 7, and 13 of the '659 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 13 are not supported by the written description in the '659 Patent specification or enabled.

- iv. “performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment”

4694. Claims 1, 7, and 13 of the '659 Patent require “performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment.” The specification provides no detail as to what a “corrective policy” or “corrective action” are or how the “corrective action” would be performed based on the “corrective policy.” *See, e.g.*, '659 Patent at 24:5-8 (“In another embodiment, the refinery, uses predictive algorithms to predict the failure of a network device and to determine appropriate corrective policy ahead of the failure.”). In its infringement contentions, however, NetFuel appears to allege that killing or restarting an alleged agent satisfies this element of claim 1 of the '659 Patent. *See, e.g.*, '659 Infringement Chart at 35 (“Corrective actions applied by software agents like CoPP or LPTS also include stopping, starting, or restarting CoPP, QoS, or LPTS processes or threads.”). But, in my view, the '659 Patent specification does not support this interpretation of the claims. The patented system is always described in terms of the agent performing the corrective action, and there is no description of a system in the '659 Patent in which an agent kills or restarts itself—or in which it is killed or restarted by the global modeler or any other entity. Moreover, there is no description in the '659 Patent specification of a system in which “corrective policy” being applied by an agent results in stopping, starting, or restarting that agent or any other agent. Even under NetFuel’s purported interpretation of the claims, which is not correct, there is no support for claims 1, 7, and



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

13. NetFuel also appears to contend that “corrective action may include modification of CoPP or LPTS policies to alter how CoPP or LPTS permits, blocks, or rate limits traffic, including by way of an EEM applet or script. ’659 Infringement Chart at 35. As described above, however, it is my view that the ’659 Patent specification also does not support this interpretation of the claims. there is no description of a system in the ’659 Patent in which blocking or rate limiting traffic are “corrective actions.”

4695. Accordingly, if claims 1, 7, and 13 of the ’659 Patent are interpreted as broadly as proposed by NetFuel in its infringement contentions, it is my view that claims 1, 7, and 13 are not supported by the written description in the ’659 Patent specification or enabled.

b. *Invalidity Under 35 U.S.C. § 112, ¶ 1—Enablement*

4696. I understand that under the first paragraph of 35 U.S.C. § 112, a patent’s specification must describe the claimed invention in such full, clear, concise, and exact terms as to enable one of ordinary skill in the art to make and use the full scope of the claimed invention. I also understand that in determining whether a patent meets this requirement, one takes into account the level of ordinary skill in the art as of the priority date of the patent. I further understand that a claim is invalid if it is not supported by an enabling disclosure.

4697. It is also my understanding that the fact that some experimentation may be required for one of ordinary skill in the art to practice the claimed invention does not necessarily mean the patent fails meet the enablement requirement. However, it is my understanding that factors that may be considered in determining whether undue experimentation would be required to make and use the claimed invention include: (1) the quantity of experimentation necessary; (2) the amount of direction or guidance disclosed in the patent; (3) the presence or absence of working examples in the patent; (4) the nature of the invention; (5) the state of the prior art; (6) the relative skill of those in the art; (7) the predictability of the art; and (8) the breadth of the claims. It is further my understanding that patents are written for persons of skill in the field of the invention. Thus, a patent need not expressly state information that skilled persons would be likely to know or could obtain.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

4698. I further understand that the full scope of the claims must be enabled, such that the scope of the claimed invention is not greater than the scope of the invention enabled by the specification. I also understand that elements of a claim may not be enabled when the inventor was not able to implement the elements at the time of filing. It is also my understanding that a patent can lack enablement of a given claim element by expressly teaching away from it.

- i. “monitoring operational parameters relating to the each thread including a per-thread utilization for the each thread”

4699. As explained above, the patent does not contain a written description of an “operational parameters” nor how to monitor an “operational parameter” or “per-thread utilization” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such a “operational parameter” or “per-thread utilization,” it is my view that the ’659 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the term “operational parameter” or “per-thread utilization.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the “operational parameter” or “per-thread utilization” lie.

- ii. “detecting if there is an abnormality in the monitored operational parameters”

4700. As described above, the patent does not contain a written description for “detecting if there is an abnormality in the monitored operational parameters” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such an “abnormality” or “operational parameters” or the method for detecting this “abnormality,” it is also my view that the ’659 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the terms “abnormality” or “operational parameters” Hence, there is no amount of experimentation that one could perform to determine how to make

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

or use the invention, because it is not clear from the claims where the boundaries of the claim terms lie.

- iii. “performing a corrective action to fix any detected abnormalities”

4701. As described above, the patent does not contain a written description for “performing a corrective action to fix any detected abnormalities” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such a “corrective action” or “detected abnormalities” or the method for detecting these “abnormalities,” it is also my view that the ’659 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the terms “corrective action” or “detected abnormalities.” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the claim terms lie.

- iv. “performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment”

4702. As described above, the patent does not contain a written description for “performing the corrective action is based on the corrective policy applied by the agent running within the first runtime environment” in the context of the claim language and under NetFuel’s apparent application in its infringement contentions. Because the patent contains no description of such an “corrective action” or “corrective policy” or the method for performing this “corrective action” based on the “corrective policy,” it is also my view that the ’659 Patent is not enabled. In particular, there is no disclosure of how one of ordinary skill in the art would make and use the claimed invention, given that there are no bounds to the terms “corrective action” or “corrective policy” Hence, there is no amount of experimentation that one could perform to determine how to make or use the invention, because it is not clear from the claims where the boundaries of the claim terms lie.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY****D. Indefiniteness**

4703. I understand that a patent specification must conclude with one or more claims particularly pointing out and distinctly claiming the subject matter that the applicant regards as his invention. I further understand that claims are indefinite if they do not reasonably apprise those of ordinary skill in the art of the applicant's intended scope of the invention when read in light of the specification. I also understand that in order to show indefiniteness, the party challenging validity must prove so by clear and convincing evidence.'730 Patent

a. *"information about a network component"*

4704. It is my opinion that claims 1, 7, and 30 of the '730 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term "obtaining information about a network component," read in light of the specification and NetFuel's infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a clear description of what "information about a network component" is such that one cannot determine the bounds of this claim term. The claim language and patent specification fail to address what "information about a network component" is in the context of NetFuel's infringement contentions and thus fail to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

b. *"test policy"*

4705. It is my opinion that claims 1, 7, and 30 of the '730 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term "test policy," read in light of the specification and NetFuel's infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a clear description of what an "test policy" is such that one cannot determine the bounds of this claim term. The claim language and patent specification fail to address what "test policy" is in the context of NetFuel's infringement contentions and thus fail to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**c. *“appropriate policy”*

4706. It is my opinion that claims 1, 7, and 30 of the '730 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term “appropriate policy,” read in light of the specification and NetFuel’s infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a clear description of what an “appropriate policy” is such that one cannot determine the bounds of this claim term. The claim language and patent specification fail to address what “appropriate policy” is in the context of NetFuel’s infringement contentions and thus fail to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

d. *“prediction algorithm”*

4707. It is my opinion that claims 1, 7, and 30 of the '730 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term “prediction algorithm” (or “predictive algorithm”), read in light of the specification and NetFuel’s infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a clear description of what a “prediction algorithm” is such that one cannot determine the bounds of this claim term. The claim language and patent specification fail to address what a “prediction algorithm” is in the context of NetFuel’s infringement contentions and thus fail to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

e. *“agent support mechanism”*

4708. It is my opinion that claims 7 and 16 of the '730 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term “agent support mechanism,” read in light of the specification and NetFuel’s infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a single mention of the term “agent support mechanism” or even “agent support.” The specification does not contain a clear description of what an “agent support mechanism” is or how to implement one “embodied in hardware” or software such that one cannot determine the bounds

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

of this claim term. The claim language and patent specification fail to address what an “agent support mechanism” is and how it would be “embodied in hardware” in the context of NetFuel’s infringement contentions and thus fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

2. ***’659 Patent***

a. *“per-thread utilization”*

4709. It is my opinion that claims 1, 7, and 13 of the ’659 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term “per-thread utilization,” read in light of the specification and NetFuel’s infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a clear description of what an “per-thread utilization” is such that one cannot determine the bounds of this claim term. The claim language and patent specification fail to address what “per-thread utilization” is in the context of NetFuel’s infringement contentions and thus fail to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

b. *“corrective policy”*

4710. It is my opinion that claims 1, 7, and 13 of the ’659 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term “corrective policy,” read in light of the specification and NetFuel’s infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a clear description of what an “corrective policy” is such that one cannot determine the bounds of this claim term. The claim language and patent specification fail to address what “corrective policy” is in the context of NetFuel’s infringement contentions and thus fail to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

c. *“corrective action”*

4711. It is my opinion that claims 1, 7, and 13 of the ’659 Patent are invalid under 35 U.S.C. § 112 as indefinite because the term “corrective action,” read in light of the specification

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

and NetFuel's infringement contentions, fails to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter. As described above, the specification does not contain a clear description of what an "corrective action" is such that one cannot determine the bounds of this claim term. The claim language and patent specification fail to address what "corrective action" is in the context of NetFuel's infringement contentions and thus fail to inform with reasonable certainty those skilled in the art about the scope of the claimed subject matter.

4712. Because of the above identified claim element, the Asserted Claims do not particularly point out and distinctly claim the subject matter that the applicant regarded at his inventions, at least under NetFuel's actual and/or apparent application of the claims.


**XIII. CONCLUSION**

4713. The opinions in this report are based upon the information presently available to me. I will amend, supplement, and/or modify any opinions and the bases for any opinions as new information becomes available or in light of Dr. Rubin's expert reports. I also reserve the right to supplement my report should the Court issue any additional claim constructions or relevant rulings. Finally, I will create demonstrative exhibits, summary charts, and the like that may be useful in presenting my opinions in further proceedings in this case.

**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**XIV. RESERVATION OF RIGHTS**

4714. My opinions are based upon the information that I have considered to date. I reserve the right to supplement or amend my opinions in response to opinions expressed by NetFuel's experts or in light of any additional evidence, testimony, or other information that may be provided to me after the date of this report, and/or that was provided to me at a late date. I explicitly reserve my right to supplement my report based on such discovery. I declare under penalty of perjury of the laws of the United States that the foregoing is true and correct.

By:   
Dr. Kevin C. Almeroth  
October 25, 2019



**HIGHLY CONFIDENTIAL - ATTORNEYS' EYES ONLY**

**CERTIFICATE OF SERVICE**

I, Frank Carlow, hereby certify that on October 25, 2019 I am serving a true and correct copy of the foregoing on counsel listed below via email.

Floyd G. Short  
Matthew R. Berry  
Steven M. Seigel  
Ryan Burningham  
SUSMAN GODFREY, L.L.P.  
1201 Third Avenue, Suite 3800  
Seattle, Washington 98101-3000  
Telephone: (206) 516-3800  
Facsimile: (206) 516-3883  
fshort@susmangodfrey.com  
mberry@susmangodfrey.com  
sseigel@susmangodfrey.com  
rburningham@susmangodfrey.com

Kalpana Srinivasan  
SUSMAN GODFREY, L.L.P.  
1900 Avenue of the Stars, Suite 1400  
Los Angeles, California 90067-6029  
Telephone: (310) 789-3100  
Facsimile: (310) 789-3150  
ksrinivasan@susmangodfrey.com

/s/ Frank Carlow

Frank Carlow, Senior IP Paralegal  
KIRKLAND & ELLIS LLP

# Appendix 1

## Kevin C. Almeroth

Professor, Department of Computer Science  
University of California  
Santa Barbara, CA 93106-5110  
(805)636-1123 (office)  
(805)893-8553 (fax)  
almeroth@cs.ucsb.edu (email)  
<http://www.cs.ucsb.edu/~almeroth> (WWW URL)

### Education

- Ph.D.** June 1997 *Georgia Institute of Technology* Computer Science  
*Dissertation Title:* Networking and System Support for the Efficient,  
Scalable Delivery of Services in Interactive Multimedia Systems  
*Minor:* Telecommunications Public Policy
- M.S.** June 1994 *Georgia Institute of Technology* Computer Science  
*Specialization:* Networking and Systems
- B.S.** June 1992 *Georgia Institute of Technology* Information and Computer Science  
**(high honors)** *Minors:* Economics, Technical Communication, American Literature

### Employment History

|                     |   |                      |
|---------------------|---|----------------------|
| Professor           | Department of Computer Science<br>University of California<br>Santa Barbara, CA | Jul 2005 -- present  |
| Associate Dean      | College of Engineering<br>University of California<br>Santa Barbara, CA         | Mar 2007 -- Aug 2009 |
| Vice Chair          | Department of Computer Science<br>University of California<br>Santa Barbara, CA | Jul 2000 -- Nov 2005 |
| Associate Professor | Department of Computer Science<br>University of California<br>Santa Barbara, CA | Jul 2001 -- Jun 2005 |

|                      |   |  |
|----------------------|---|--|
| Assistant Professor  | Department of Computer Science<br>University of California<br>Santa Barbara, CA           | Jul 1997 -- Jun 2001   |
| Graduate Researcher  | Broadband Telecommunications Center<br>Georgia Center for Adv Telecom Tech<br>Atlanta, GA | Sep 1996--Jun 1997   |
| Graduate Intern      | IBM<br>T.J. Watson Research Labs<br>Hawthorne, NY   | Jun 1995--Sep 1995   |
| Support Specialist   | Office of Information Technology<br>Georgia Institute of Technology<br>Atlanta, GA        | Sep 1995--Jun 1997   |
| Research Assistant   | College of Computing<br>Georgia Institute of Technology<br>Atlanta, GA                    | Jan 1994--Mar 1994   |
| Graduate Intern      | Hitachi Telecommunications<br>Norcross, GA  | Jun 1992--Sep 1992   |
| Undergraduate Intern | IBM<br>Research Triangle Park, NC   | Jun 1989--Sep 1989<br>Jun 1990--Sep 1990<br>Mar 1991--Sep 1991 |

## Industry Technical Advising

|                                       |  |                         |
|---------------------------------------|--|-------------------------|
| Board of Directors                    | <a href="#"><u>The New Media Studio</u></a><br>Santa Barbara, CA | Nov 2006 --<br>present  |
| Co-Founder &<br>Chairman of the Board | Santa Barbara Labs, LLC<br>Santa Barbara, CA                     | Sep 2007 --<br>Dec 2009 |
| Board of Advisors                     | Techknowledge Point<br>Santa Barbara, CA                         | May 2001 --<br>Dec 2007 |
| Technical Advisory Board              | Occam Networks, Inc.<br>Santa Barbara, CA                        | May 2000 --<br>Dec 2010 |
| Board of Advisors                     | Airplay Inc.<br>San Francisco, CA                                | Jun 2005 --<br>Aug 2009 |
| Consultant                            | Lockheed Martin Corporation<br>San Jose, CA                      | Nov 1999 --<br>Jun 2009 |
| Board of Advisors                     | Santa Barbara Technology Group<br>Santa Barbara, CA              | Sep 2000 --<br>Dec 2004 |
| Board of Directors                    | Virtual Bandwidth, Inc.<br>Santa Barbara, CA                     | Nov 2000 --<br>Jun 2001 |

|   |  |                         |
|---|--|-------------------------|
| Board of Advisors &<br>Affiliated Scientist | Digital Fountain<br>San Francisco, CA                    | Jan 2000 --<br>Dec 2001 |
| Senior Technologist                         | IP Multicast Initiative, Stardust Forums<br>Campbell, CA | Jun 1998 --<br>Dec 2000 |

## I. Teaching

### A. Courses Taught

|             |  |   |
|-------------|--|---|
| CS<br>176A  | Intro to Computer<br>Communication<br>Networks | Fall 1997, Fall 1998, Fall 2002, Fall 2003, Fall 2004, Spring 2005,<br>Spring 2006, Spring 2007, Spring 2008, Fall 2008, Fall 2009, Fall<br>2010, Fall 2011, Fall 2012, Fall 2013, Fall 2014, Spring 2017,<br>Spring 2018 |
| CS<br>176B  | Network Computing                              | Winter 2000, Winter 2001, Winter 2002, Winter 2012, Winter 2014,<br>Winter 2015, Winter 2018, Winter 2019   |
| MAT<br>201B | Media Networks and<br>Services                 | Fall 1999, Fall 2000, Fall 2001, Fall 2003  |
| CS<br>276   | Distributed Computing<br>and Computer Networks | Winter 1999, Spring 2000, Fall 2002, Fall 2005, Fall 2018   |
| CS<br>290I  | Networking for<br>Multimedia Systems           | Winter 1998, Spring 1999, Fall 2004, Winter 2010  |
| CS<br>595N  | Technology and Society                         | Winter 2005, Fall 2005, Spring 2006, Fall 2006, Spring 2007, Fall<br>2007, Spring 2008, Fall 2008, Spring 2009  |
| CS<br>595N  | Economic Systems<br>Seminar                    | Winter 2004, Spring 2004, Winter 2005, Spring 2005  |
| CS<br>595N  | Networking Seminar                             | Winter 1999, Fall 1999, Winter 2003, Winter 2019  |
| CS<br>595N  | Wireless Networking &<br>Multimedia Seminar    | Fall 2000   |
| CS<br>595I  | Systems Design and<br>Implementation Seminar   | Fall 1999, Fall 2000, Winter 2001, Spring 2001, Winter 2002,<br>Spring 2002   |

### B. Other Teaching Experience

- *The Evolution of Advanced Networking Services: From the ARPAnet to Internet2*, Instructor, Summer 2001. Short course taught at Escuela de Ciencias Informatica (ECI) sponsored by the Universidad de Buenos Aires.
- *Johns Hopkins Center for Talented Youth*, Instructor, Summer 1994. CTY is a program to teach gifted high school students the fundamentals of computer science.
- *Georgia Institute of Technology*, Graduate Teaching Assistant, Sep 1994--Sep 1996. Worked as a TA for

12 quarters teaching 7 different courses (4 undergraduate and 3 graduate).

### C. Ph.D. Students Advised [14 graduated]

14. Daniel Havey  
Research Area: *Throughput and Delay on the Packet Switched Internet*  
Date Graduated: Winter 2015  
First Position: Microsoft
13. Lara Deek (co-advised with E. Belding)  
Research Area: *Resource-Efficient Wireless Systems for Emerging Wireless Networks*  
Date Graduated: Summer 2014  
First Position: Post Doc, UIUC
12. Mike Wittie  
Research Area: *Towards Sustained Scalability of Communication Networks*  
Date Graduated: Summer 2011  
First Position: Assistant Professor, Montana State University
11. Allan Knight  
Research Area: *Supporting Integration of Educational Technologies and Research of Their Effects on Learning*  
Date Graduated: Summer 2009  
First Position: Research Scientist, Citrix Online
10. Hangjin Zhang  
Research Area: *Towards Blended Learning: Educational Technology to Improve and Assess Teaching and Learning*  
Date Graduated: Spring 2009  
First Position: Microsoft
9. Gayatri Swamynathan  
Dissertation Title: *Towards Reliable Reputations for Distributed Applications*  
Date Graduated: Spring 2008  
First Position: Zynga
8. Amit Jardosh (co-advised with E. Belding)  
Dissertation Title: *Adaptive Large-Scale Wireless Networks: Measurements, Protocol Designs, and Simulation Studies*  
Date Graduated: Fall 2007  
First Position: Yahoo!
7. Khaled Harras  
Dissertation Title: *Protocol and Architectural Challenges in Delay and Disruption Tolerant Networks*  
Date Graduated: Summer 2007  
First Position: Assistant Professor, Carnegie Mellon University
6. Krishna Ramachandran (co-advised with E. Belding)  
Dissertation Title: *Design, Deployment, and Management of High-Capacity Wireless Mesh Networks*  
Date Graduated: Winter 2006  
First Position: Research Scientist, Citrix Online
5. Robert Chalmers  
Dissertation Title: *Improving Device Mobility with Intelligence at the Network Edge*  
Date Graduated: Summer 2004  
First Position: President and CEO, Limbo.net

4. Prashant Rajvaidya  
 Dissertation Title: *Achieving Robust and Secure Deployment of Multicast*  
 Date Graduated: Spring 2004  
 First Position: President and CTO, Mosaic Networking
3. Sami Rollins  
 Dissertation Title: *Overcoming Resource Constraints to Enable Content Exchange Applications in Next-Generation Environments*  
 Date Graduated: Spring 2003  
 First Position: Assistant Professor, Mount Holyoke College
2. Srinivasan Jagannathan  
 Dissertation Title: *Multicast Tree-Based Congestion Control and Topology Management*  
 Date Graduated: Spring 2003  
 First Position: Consultant, Kelly & Associates
1. Kamil Sarac  
 Dissertation Title: *Supporting a Robust Multicast Service in the Global Infrastructure*  
 Date Graduated: Spring 2002  
 First Position: Assistant Professor, UT-Dallas

#### **D. M.S. Students Advised (Thesis/Project Option) [19 graduated]**

19. Neer Shey  
 Research Area: *Analyzing Content Distribution Through Opportunistic Contact for Smart Cellular Phones*  
 Date Graduated: Spring 2010
18. Camilla Fiorese  
 Research Area: *Analysis of a Pure Rate-Based Congestion Control Algorithm*  
 Date Graduated: Summer 2009
17. Brian Weiner  
 Research Area: *Multi-Socket TCP: A Simple Approach to Improve Performance of Real-Time Applications over TCP*  
 Date Graduated: Fall 2007
16. Avijit Sen Mazumder  
 Research Area: *Facilitating Robust Multicast Group Management*  
 Date Graduated: Fall 2005
15. Rishi Matthew  
 Thesis Title: *Providing Seamless Access to Multimedia Content on Heterogeneous Platforms*  
 Date Graduated: Summer 2004
14. Camden Ho  
 Research Area: *Tools and Techniques for Wireless Network Management*  
 Date Graduated: Spring 2004
13. Amit Jardosh (co-advised with E. Belding)  
 Research Area: *Realistic Environment Models for Mobile Network Evaluation*  
 Date Graduated: Spring 2004
12. Nitin Solanki  
 Research Area: *SongWand: A Wireless Barcode Scanner Using Bluetooth Technology*  
 Date Graduated: Winter 2004
11. Vrishali Wagle (co-advised with E. Belding)  
 Research Area: *An Ontology-Based Service Discovery Mechanism*  
 Date Graduated: Winter 2004

10. Uday Mohan  
Thesis Title: *Scalable Service Discovery in Mobile Ad hoc Networks*  
Date Graduated: Spring 2003
9. Krishna Ramachandran  
Thesis Title: *Ubiquitous Multicast*  
Date Graduated: Spring 2003
8. John Slonaker  
Thesis Title: *Inductive Loop Signature Acquisition Techniques*  
Date Graduated: Spring 2002
7. Mohammad Battah  
Thesis Title: *Dedicated Short-Range Communications Intelligent Transportation Systems Protocol (DSRC-ITS)*  
Date Graduated: Spring 2002
6. Kevin Vogel  
Thesis Title: *Integrating E-Commerce Applications into Existing Business Infrastructures*  
Date Graduated: Spring 2001
5. Sami Rollins  
Thesis Title: *Audio Xml: Aural Interaction with XML Documents*  
Date Graduated: Winter 2000
4. Andy Davis  
Thesis Title: *Stream Scheduling for Data Servers in a Scalable Interactive TV System*  
Date Graduated: Spring 1999
3. David Makofske  
Thesis Title: *MHealth: A Real-Time Graphical Multicast Monitoring Tool*  
Date Graduated: Winter 1999
2. Prashant Rajvaidya  
Thesis Title: *MANTRA: Router-Based Monitoring and Analysis of Multicast Traffic*  
Date Graduated: Winter 1999
1. Alex DeCastro (co-advised with Yuan-Fang Wang)  
Thesis Title: *Web-Based Collaborative 3D Modeling*  
Date Graduated: Winter 1998

## **E. Teaching Awards**

2006-2007 UCSB Academic Senate Distinguished Teaching Award  
2004-2005 Computer Science Outstanding Faculty Member  
2000-2001 UCSB Spotlight on Excellence Award  
1999-2000 Computer Science Outstanding Faculty Member (co-recipient)  
1998-1999 Computer Science Outstanding Faculty Member (co-recipient)  
1997-1998 Computer Science Outstanding Faculty Member

## **II. Research**

### **A. Journal Papers, Magazine Articles, Books, and Book Chapters**



62. L. Deek, E. Garcia-Villegas, E. Belding, S.J. Lee, and K. Almeroth, "[A Practical Framework for 802.11 MIMO Rate Adaptation](#)," *Computer Networks*, vol. 83, num. 6, pp. 332-348, June 2015.
61. L. Deek, E. Garcia-Villegas, E. Belding, S.J. Lee, and K. Almeroth, "[Intelligent Channel Bonding in 802.11n WLANs](#)," *IEEE Transactions on Mobile Computing*, vol. 13, num. 6, pp. 1242-1255, June 2014.
60. H. Zhang and K. Almeroth, "[Alternatives for Monitoring and Limiting Network Access to Students in Network-Connected Classrooms](#)," *Journal of Interactive Learning Research (JILR)*, vol. 24, num. 3, pp. 237-265, July 2013.
59. M. Tavakolifard and K. Almeroth, "[A Taxonomy to Express Open Challenges in Trust and Reputation Systems](#)," *Journal of Communications*, vol. 7, num. 7, pp. 538-551, July 2012.
58. M. Tavakolifard and K. Almeroth, "[Social Computing: An Intersection of Recommender Systems, Trust/Reputation Systems, and Social Networks](#)," *IEEE Network*, vol. 26, num. 4, pp. 53-58, July/August 2012.
57. M. Tavakolifard, K. Almeroth, and P. Ozturk, "[Subjectivity Handling of Ratings for Trust and Reputation Systems: An Abductive Reasoning Approach](#)," *International Journal of Digital Content Technology and its Applications (JDCTA)*, vol. 5, num. 11, pp. 359-377, November 2011.
56. R. Raghavendra, P. Acharya, E. Belding and K. Almeroth, "[MeshMon: A Multi-Tiered Framework for Wireless Mesh Network Monitoring](#)," *Wireless Communications and Mobile Computing (WCMC) Journal*, vol. 11, num. 8, pp. 1182-1196, August 2011.
55. A. Knight and K. Almeroth, "[Automatic Plagiarism Detection with PAIRwise 2.0](#)," *Journal of Interactive Learning Research (JILR)*, vol. 22, num. 3, pp. 379-400, July 2011.
54. V. Kone, M. Zheleva, M. Wittie, B. Zhao, E. Belding, H. Zheng, and K. Almeroth, "[AirLab: Consistency, Fidelity and Privacy in Wireless Measurements](#)," *ACM Computer Communications Review*, vol. 41, num. 1, pp. 60-65, January 2011.
53. G. Swamynathan, K. Almeroth, and B. Zhao, "[The Design of a Reliable Reputation System](#)," *Electronic Commerce Research Journal*, vol. 10, num. 3-4, pp. 239-270, December 2010.
52. P. Acharya, A. Sharma, E. Belding, K. Almeroth and K. Papagiannaki, "[Rate Adaptation in Congested Wireless Networks through Real-Time Measurements](#)," *IEEE Transactions on Mobile Computing*, vol. 9, num. 11, pp. 1535-1550, November 2010.
51. R. Raghavendra, E. Belding, K. Papagiannaki, and K. Almeroth, "[Unwanted Link Layer Traffic in Large IEEE 802.11 Wireless Networks](#)," *IEEE Transactions on Mobile Computing*, vol. 9, num. 9, pp. 1212-1225, September 2010.
50. H. Zhang and K. Almeroth, "[Moodog: Tracking Student Activity in Online Course Management Systems](#)," *Journal of Interactive Learning Research (JILR)*, vol. 21, num. 3, pp. 407-429, July 2010.
49. R. Chertov and K. Almeroth, "[Qualitative Comparison of Link Shaping Techniques](#)," *International Journal of Communication Networks and Distributed Systems*, vol. 5, num. 1/2, pp. 109-129, July 2010.
48. A. Knight and K. Almeroth, "[Fast Caption Alignment for Automatic Indexing of Audio](#)," *International Journal of Multimedia Data Engineering and Management*, vol. 1, num. 2, pp. 1-17, April-June 2010.

47. K. Harras and K. Almeroth, "[Scheduling Messengers in Disconnected Clustered Mobile Networks](#)," Ad Hoc & Sensor Wireless Networks, vol. 9, num. 3-4, pp. 275-304, March-April 2010.
46. A. Jardosh, K. Papagiannaki, E. Belding, K. Almeroth, G. Iannaccone, and B. Vinnakota, "[Green WLANs: On-Demand WLAN Infrastructures](#)," ACM Journal on Mobile Networks and Applications (MONET), vol. 14, num. 6, pp. 798-814, December 2009.
45. M. Wittie, K. Harras, K. Almeroth, and E. Belding, "[On the Implications of Routing Metric Staleness in Delay Tolerant Networks](#)," Computer Communications Special Issue on Delay and Disruption Tolerant Networking, vol. 32, num. 16, pp. 1699-1709, October 2009.
44. K. Harras, L. Deek, C. Holman, and K. Almeroth, "[DBS-IC: An Adaptive Data Bundling System for Intermittent Connectivity](#)," Computer Communications Special Issue on Delay and Disruption Tolerant Networking, vol. 32, num. 16, pp. 1687-1698, October 2009.
43. S. Karpinski, E. Belding, K. Almeroth, and J. Gilbert, "[Linear Representations of Network Traffic](#)," ACM Journal on Mobile Networks and Applications (MONET), vol. 14, num. 4, pp. 368-386, August 2009.
42. K. Harras and K. Almeroth, "[Controlled Flooding in Disconnected Sparse Mobile Networks](#)," Wireless Communications and Mobile Computing (WCMC) Journal, vol. 9, num. 1, pp. 21-33, January 2009.
41. R. Mayer, A. Stull, K. DeLeeuw, K. Almeroth, B. Bimber, D. Chun, M. Bulger, J. Campbell, A. Knight, and H. Zhang, "[Clickers in College Classrooms: Fostering Learning with Questioning Methods in Large Lecture Classes](#)," Contemporary Educational Psychology, vol. 34, num. 1, pp. 51-57, January 2009.
40. A. Knight, K. Almeroth, and B. Bimber, "[Design, Implementation and Deployment of PAIRwise](#)," Journal of Interactive Learning Research (JILR), vol. 19, num. 3, pp. 489-508, July 2008.
39. A. Garyfalos and K. Almeroth, "[Coupons: A Multilevel Incentive Scheme for Information Dissemination in Mobile Networks](#)," IEEE Transactions on Mobile Computing, vol. 7, num. 6, pp. 792-804, June 2008.
38. I. Sheriff, K. Ramachandran, E. Belding, and K. Almeroth, "[A Multi-Radio 802.11 Mesh Network Architecture](#)," ACM Journal on Mobile Networks and Applications (MONET), vol. 13, num. 1-2, pp. 132-146, April 2008.
37. M. Bulger, R. Mayer, K. Almeroth, and S. Blau, "[Measuring Learner Engagement in Computer-Equipped College Classrooms](#)," Journal of Educational Multimedia and Hypermedia, vol. 17, num. 2, pp. 129-143, April 2008.
36. G. Swamynathan, B. Zhao, and K. Almeroth, "[Exploring the Feasibility of Proactive Reputations](#)," Concurrency and Computation: Practice and Experience, vol. 20, num. 2, pp. 155-166, February 2008.
35. G. Swamynathan, B. Zhao, K. Almeroth, and H. Zheng, "[Globally Decoupled Reputations for Large Distributed Networks](#)," Advances in Multimedia, vol. 2007, pp. 1-14, 2007.
34. R. Mayer, A. Stull, J. Campbell, K. Almeroth, B. Bimber, D. Chun and A. Knight, "[Overestimation Bias in Self-reported SAT Scores](#)," Educational Psychology Review, vol. 19, num. 4, pp. 443-454, December 2007.
33. P. Namburi, K. Sarac and K. Almeroth, "[Practical Utilities for Monitoring Multicast Service Availability](#)," Computer Communications Special Issue on Monitoring and Measurement of IP

Networks, vol. 29, num. 10, pp. 1675-1686, June 2006.

32. R. Chalmers, G. Krishnamurthi and K. Almeroth, "[Enabling Intelligent Handovers in Heterogeneous Wireless Networks](#)," ACM Journal on Mobile Networks and Applications (MONET), vol. 11, num. 2, pp. 215-227, April 2006.
31. H. Lundgren, K. Ramachandran, E. Belding, K. Almeroth, M. Benny, A. Hewatt, A. Touma and A. Jardosh, "[Experience from the Design, Deployment and Usage of the UCSB MeshNet Testbed](#)," IEEE Wireless Communications, vol. 13, num. 2, pp. 18-29, April 2006.
30. R. Mayer, K. Almeroth, B. Bimber, D. Chun, A. Knight and A. Campbell, "[Technology Comes to College: Understanding the Cognitive Consequences of Infusing Technology in College Classrooms](#)," Educational Technology, vol. 46, num. 2, pp. 48-53, March-April 2006.
29. A. Garyfalos and K. Almeroth, "[A Flexible Overlay Architecture for Mobile IPv6 Multicast](#)," Journal on Selected Areas in Communications (JSAC) Special Issue on Wireless Overlay Networks Based on Mobile IPv6, vol. 23, num. 11, pp. 2194-2205, November 2005.
28. K. Sarac and K. Almeroth, "[Monitoring IP Multicast in the Internet: Recent Advances and Ongoing Challenges](#)," IEEE Communications, vol. 43, num. 10, pp. 85-91, October 2005.
27. K. Sarac and K. Almeroth, "[Application Layer Reachability Monitoring for IP Multicast](#)," Computer Networks, vol. 48, num. 2, pp. 195-213, June 2005.
26. A. Jardosh, E. Belding, K. Almeroth and S. Suri, "[Real-world Environment Models for Mobile Network Evaluation](#)," Journal on Selected Areas in Communications Special Issue on Wireless Ad hoc Networks, vol. 23, num. 3, pp. 622-632, March 2005.
25. S. Rollins and K. Almeroth, "[Evaluating Performance Tradeoffs in a One-to-Many Peer Content Distribution Architecture](#)," Journal of Internet Technology, vol. 5, num. 4, pp. 373-387, Fall 2004.
24. K. Sarac and K. Almeroth, "[Tracetree: A Scalable Mechanism to Discover Multicast Tree Topologies in the Network](#)," IEEE/ACM Transactions on Networking, vol. 12, num. 5, pp. 795-808, October 2004.
23. K. Sarac and K. Almeroth, "[A Distributed Approach for Monitoring Multicast Service Availability](#)," Journal of Network and Systems Management, vol. 12, num. 3, pp. 327-348, September 2004.
22. P. Rajvaidya, K. Ramachandran and K. Almeroth, "[Managing and Securing the Global Multicast Infrastructure](#)," Journal of Network and Systems Management, vol. 12, num. 3, pp. 297-326, September 2004.
21. P. Rajvaidya and K. Almeroth, "[Multicast Routing Instabilities](#)," IEEE Internet Computing, vol. 8, num. 5, pp. 42-49, September/October 2004.
20. D. Johnson, R. Patton, B. Bimber, K. Almeroth and G. Michaels, "[Technology and Plagiarism in the University: Brief Report of a Trial in Detecting Cheating](#)," Association for the Advancement of Computing in Education (AACE) Journal, vol. 12, num. 3, pp. 281-299, Summer 2004.
19. R. Chalmers and K. Almeroth, "[A Security Architecture for Mobility-Related Services](#)," Journal of Wireless Personal Communications, vol. 29, num. 3, pp. 247-261, June 2004.
18. B. Stiller, K. Almeroth, J. Altmann, L. McKnight, and M. Ott, "[Pricing for Content in the Internet](#)," Computer Communications, vol. 27, num. 6, pp. 522-528, April 2004.

17. S. Rollins and K. Almeroth, "[Lessons Learned Deploying a Digital Classroom](#)," *Journal of Interactive Learning Research (JILR)*, vol. 15, num. 2, pp. 169-185, April 2004.
16. S. Jagannathan and K. Almeroth, "[A Dynamic Pricing Scheme for E-Content at Multiple Levels-of-Service](#)," *Computer Communications*, vol. 27, num. 4, pp. 374-385, March 2004.
15. K. Almeroth, "[Using Satellite Links in the Delivery of Terrestrial Multicast Traffic](#)," *Internetworking and Computing over Satellites*, Kluwer Academic Publishers, 2003.
14. R. Chalmers and K. Almeroth, "[On the Topology of Multicast Trees](#)," *IEEE/ACM Transactions on Networking*, vol. 11, num. 1, pp. 153-165, January 2003.
13. S. Jagannathan, J. Nayak, K. Almeroth, and M. Hofmann, "[On Pricing Algorithms for Batched Content Delivery Systems](#)," *Electronic Commerce Research and Applications Journal*, vol. 1, num. 3-4, pp. 264-280, Fall 2002.
12. D. Makofske and K. Almeroth, "[Multicast Sockets: Practical Guide for Programmers](#)," *Morgan Kaufmann Publishers*, November 2002.
11. S. Jagannathan and K. Almeroth, "[Price Issues in Delivering E-Content On-Demand](#)," *ACM Sigecom Exchanges*, vol. 3, num. 2, pp. 18-27, May 2002.
10. D. Makofske and K. Almeroth, "[From Television to Internet Video-on-Demand: Techniques and Tools for VCR-Style Interactivity](#)," *Software: Practice and Experience*, vol. 31, num. 8, pp. 781-801, July 2001.
9. K. Sarac and K. Almeroth, "[Supporting Multicast Deployment Efforts: A Survey of Tools for Multicast Monitoring](#)," *Journal on High Speed Networking, Special Issue on Management of Multimedia Networking*, vol. 9, num. 3/4, pp. 191-211, March 2001.
8. K. Almeroth, "[Adaptive, Workload-Dependent Scheduling for Large-Scale Content Delivery Systems](#)," *Transactions on Circuits and Systems for Video Technology, Special Issue on Streaming Video*, vol. 11, num. 3, pp. 426-439, March 2001.
7. D. Makofske and K. Almeroth, "[Real-Time Multicast Tree Visualization and Monitoring](#)," *Software: Practice and Experience*, vol. 30, num. 9, pp. 1047-1065, July 2000.
6. M. Ammar, K. Almeroth, R. Clark and Z. Fei, "Multicast Delivery of WWW Pages," *Electronic Commerce Technology Trends: Challenges and Opportunities*, IBM Press, February 2000.
5. K. Almeroth, "[The Evolution of Multicast: From the MBone to Inter-Domain Multicast to Internet2 Deployment](#)," *IEEE Network Special Issue on Multicasting*, vol. 10, num. 1, pp. 10-20, January/February 2000.
4. K. Almeroth and M. Ammar, "[An Alternative Paradigm for Scalable On-Demand Applications: Evaluating and Deploying the Interactive Multimedia Jukebox](#)," *IEEE Transactions on Knowledge and Data Engineering Special Issue on Web Technologies*, vol. 11, num. 4, pp 658-672, July/August 1999.
3. K. Almeroth and M. Ammar, "[The Interactive Multimedia Jukebox \(IMJ\): A New Paradigm for the On-Demand Delivery of Audio/Video](#)," *Computer Networks and ISDN Systems*, vol. 30, no. 1, April 1998.
2. K. Almeroth and M. Ammar, "[Multicast Group Behavior in the Internet's Multicast Backbone \(MBone\)](#)," *IEEE Communications*, vol. 35, no. 6, pp. 124-129, June 1997.

1. K. Almeroth and M. Ammar, "[On the Use of Multicast Delivery to Provide a Scalable and Interactive Video-on-Demand Service](#)," *Journal on Selected Areas of Communication (JSAC)*, vol. 14, no. 6, pp. 1110-1122, August 1996.

## B. Conference Papers with Proceedings (refereed)

89. D. Havey and K. Almeroth, "[Active Sense Queue Management \(ASQM\)](#)," *IFIP Networking Conference*, Toulouse, FRANCE, May 2015.
88. L. Deek, E. Garcia-Villegas, E. Belding, S.J. Lee, and K. Almeroth, "[Joint Rate and Channel Width Adaptation in 802.11 MIMO Wireless Networks](#)," *IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, New Orleans, LA, USA, June 2013.
87. D. Havey and K. Almeroth, "[Fast Wireless Protocol: A Network Stack Design for Wireless Transmission](#)," *IFIP Networking Conference*, Brooklyn, New York, USA, May 2013.
86. M. Tavakolifard, J. Gulla, K. Almeroth, J. Ingvaldsen, G. Nygreen, and E. Berg, "[Tailored News in the Palm of Your HAND: A Multi-Perspective Transparent Approach to News Recommendation](#)," *Demo Track at the International World Wide Web Conference (WWW)*, Rio de Janeiro, BRAZIL, May 2013.
85. S. Patterson, M. Wittie, K. Almeroth, and B. Bamieh, "[Network Optimization with Dynamic Demands and Link Prices](#)," *Allerton Conference*, Monticello, Illinois, USA, October 2012.
84. D. Havey, R. Chertov, and K. Almeroth, "[Receiver Driven Rate Adaptation](#)," *ACM Multimedia Systems Conference (MMSys)*, Chapel Hill, North Carolina, USA, February 2012.
83. M. Tavakolifard and K. Almeroth, "[Trust 2.0: Who to Believe in the Flood of Online Data?](#)" *International Conference on Computing, Networking and Communications (ICNC)*, Maui, Hawaii, USA, January 2012.
82. L. Deek, E. Garcia-Villegas, E. Belding, S.J. Lee, and K. Almeroth, "[The Impact of Channel Bonding on 802.11n Network Management](#)," *ACM CoNEXT*, Tokyo, JAPAN, December 2011.
81. L. Deek, X. Zhou, K. Almeroth, and H. Zheng, "[To Preempt or Not: Tackling Bid and Time-based Cheating in Online Spectrum Auctions](#)," *IEEE Infocom*, Shanghai, CHINA, April 2011.
80. M. Wittie, V. Pejovic, L. Deek, K. Almeroth, and B. Zhao, "[Exploiting Locality of Interest in Online Social Networks](#)," *ACM CoNEXT*, Philadelphia, Pennsylvania, USA, November 2010.
79. R. Chertov and K. Almeroth, "[Using BGP in a Satellite-Based Challenged Network Environment](#)," *IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Boston, Massachusetts, USA, June 2010.
78. R. Chertov, D. Havey and K. Almeroth, "[MSET: A Mobility Satellite Emulation Testbed](#)," *IEEE Infocom*, San Diego, California, USA, March 2010.
77. B. Stone-Gross, A. Moser, C. Kruegel, E. Kirda, and K. Almeroth, "[FIRE: FInding Rogue nEtworks](#)," *Annual Computer Security Applications Conference (ACSAC)*, Honolulu, Hawaii, USA, December 2009.
76. M. Wittie, K. Almeroth, E. Belding, I. Rimac, and V. Hilt, "[Internet Service in Developing Regions](#)



[Through Network Coding](#)," *IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Rome, ITALY, June 2009.

75. R. Chertov and K. Almeroth, "[High-Fidelity Link Shaping](#)," *International Conference on Testbeds and Research Infrastructures for the Development of Networks and Communities (TRIDENTCOM)*, Washington DC, USA, April 2009.
74. L. Deek, K. Almeroth, M. Wittie, and K. Harras, "[Exploiting Parallel Networks Using Dynamic Channel Scheduling](#)," *International Wireless Internet Conference (WICON)*, Maui, Hawaii, USA, November 2008.
73. D. Havey, E. Barlas, R. Chertov, K. Almeroth, and E. Belding, "[A Satellite Mobility Model for QUALNET Network Simulations](#)," *IEEE Military Communications Conference (MILCOM)*, San Diego, California, USA, November 2008.
72. J. Kayfetz and K. Almeroth, "[Creating Innovative Writing Instruction for Computer Science Graduate Students](#)," *ASEE/IEEE Frontiers in Education (FIE) Conference*, Saratoga Springs, New York, USA, October 2008.
71. G. Swamynathan, B. Zhao, K. Almeroth, and S. Rao, "[Towards Reliable Reputations for Dynamic Networked Systems](#)," *IEEE International Symposium on Reliable Distributed Systems (SRDS)*, Napoli, ITALY, October 2008.
70. B. Stone-Gross, D. Sigal, R. Cohn, J. Morse, K. Almeroth, and C. Krugel, "[VeriKey: A Dynamic Certificate Verification System for Public Key Exchanges](#)," *Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA)*, Paris, FRANCE, July 2008.
69. P. Acharya, A. Sharma, E. Belding, K. Almeroth, K. Papagiannaki, "[Congestion-Aware Rate Adaptation in Wireless Networks: A Measurement-Driven Approach](#)," *IEEE Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, San Francisco, California, USA, June 2008.
68. A. Jardosh, P. Suwannat, T. Hollerer, E. Belding, and K. Almeroth, "[SCUBA: Focus and Context for Real-time Mesh Network Health Diagnosis](#)," *Passive and Active Measurement Conference (PAM)*, Cleveland, Ohio, USA, April 2008.
67. B. Stone-Gross, C. Wilson, K. Almeroth, E. Belding, H. Zheng, K. Papagiannaki, "[Malware in IEEE 802.11 Wireless Networks](#)," *Passive and Active Measurement Conference (PAM)*, Cleveland, Ohio, USA, April 2008.
66. R. Raghavendra, E. Belding, K. Papagiannaki, and K. Almeroth, "[Understanding Handoffs in Large IEEE 802.11 Wireless Networks](#)," *Internet Measurement Conference (IMC)*, San Diego, California, USA, October 2007.
65. M. Wittie, B. Stone-Gross, K. Almeroth and E. Belding, "[MIST: Cellular Data Network Measurement for Mobile Applications](#)," *IEEE International Conference on Broadband Communications, Networks, and Systems (BroadNets)*, Raleigh, North Carolina, USA, September 2007.
64. S. Karpinski, E. Belding, K. Almeroth, "[Wireless Traffic: The Failure of CBR Modeling](#)," *IEEE International Conference on Broadband Communications, Networks, and Systems (BroadNets)*, Raleigh, North Carolina, USA, September 2007.
63. A. Knight, K. Almeroth, H. Zhang, R. Mayer, and K. DeLeeuw, "[Data Cafe: A Dining Car Approach to Educational Research Data Management and Distribution](#)," *World Conference on Educational*

*Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Vancouver, CANADA, June 2007.

62. H. Zhang, K. Almeroth, A. Knight, M. Bulger, and R. Mayer, "[Moodog: Tracking Students' Online Learning Activities](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Vancouver, CANADA, June 2007.
61. M. Bulger, K. Almeroth, R. Mayer, D. Chun, A. Knight, H. Collins, "[Effects of Instructor Engagement on Student Use of a Course Management System](#)," Association for Psychological Science (APS) Annual Conference, Washington DC, USA, May 2007.
60. R. Mayer, A. Stull, K. Almeroth, B. Bimber, D. Chun, M. Bulger, J. Campbell, Allan Knight, and H. Zhang, "[Using Technology-Based Methods to Foster Learning in Large Lecture Classes: Evidence for the Pedagogic Value of Clickers](#)," *American Educational Research Association (AERA) Annual Conference*, Chicago, Illinois, USA, April 2007.
59. K. Ramachandran, I. Sheriff, E. Belding, and K. Almeroth, "[Routing Stability in Static Wireless Mesh Networks](#)," *Passive and Active Measurement Conference (PAM)*, Louvain-la-neuve, BELGIUM, April 2007.
58. G. Swamynathan, T. Close, S. Banerjee, R. McGeer, B. Zhao, and K. Almeroth, "[Scalable Access Control For Web Services](#)," *International Conference on Creating, Connecting and Collaborating through Computing (C5)*, Kyoto, JAPAN, January 2007.
57. A. Knight, M. Bulger, K. Almeroth, and H. Zhang, "[Is Learning Really a Phone Call Away? Knowledge Transfer in Mobile Learning](#)," *World Conference on Mobile Learning (mLearn)*, Banff, Alberta, CANADA, October 2006.
56. J. Kurian, K. Sarac, and K. Almeroth, "[Defending Network-Based Services Against Denial of Service Attacks](#)," *International Conference on Computer Communication and Networks (IC3N)*, Arlington, Virginia, USA, October 2006.
55. A. Jardosh, K. Sanzgiri, E. Belding and K. Almeroth, "[IQU: Practical Queue-Based User Association Management for WLANs--Case Studies, Architecture, and Implementation](#)," *ACM Mobicom*, Marina del Rey, California, USA, September 2006.
54. C. Holman, K. Harras, and K. Almeroth, "[A Proactive Data Bundling System for Intermittent Mobile Connections](#)," *IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, Reston, Virginia, USA, September 2006.
53. G. Banks, M. Cova, V. Felmetzger, K. Almeroth, R. Kemmerer and G. Vigna, "[SNOOZE: toward a Stateful NetwOrk prOtocol fuzZEer](#)," *Information Security Conference (ISC)*, Samos Island, GREECE, September 2006.
52. K. Harras and K. Almeroth, "[Inter-Regional Messenger Scheduling in Delay Tolerant Mobile Networks](#)," *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Niagara Falls, New York, USA, June 2006.
51. M. Bulger, R. Mayer, and K. Almeroth, "[Engaged By Design: Using Simulation to Promote Active Learning](#)," **Outstanding Paper** at the *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Orlando, Florida, USA, June 2006.
50. A. Knight, K. Almeroth, R. Mayer, D. Chun, and B. Bimber, "[Observations and Recommendations for Using Technology to Extend Interaction](#)," *World Conference on Educational Multimedia, Hypermedia*

& *Telecommunications (ED MEDIA)*, Orlando, Florida, USA, June 2006.

49. H. Zhang, and K. Almeroth, "[A Simple Classroom Network Access Control System](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Orlando, Florida, USA, June 2006.
48. K. Harras and K. Almeroth, "[Transport Layer Issues in Delay Tolerant Mobile Networks](#)," *IFIP Networking Conference*, Coimbra, PORTUGAL, May 2006.
47. R. Mayer, A. Stull, J. Campbell, K. Almeroth, B. Bimber, D. Chun and A. Knight, "[Some Shortcomings of Soliciting Students' Self-Reported SAT Scores](#)," *American Educational Research Association (AERA) Annual Conference*, San Francisco, California, USA, April 2006.
46. K. Ramachandran, E. Belding, K. Almeroth, and M. Buddhikot, "[Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks](#)," *IEEE Infocom*, Barcelona, SPAIN, April 2006.
45. A. Jardosh, K. Ramachandran, K. Almeroth, and E. Belding, "[Understanding Congestion in IEEE 802.11b Wireless Networks](#)," *ACM/USENIX Internet Measurement Conference (IMC)*, Berkeley, California, USA, October 2005.
44. H. Zhang, K. Almeroth and M. Bulger, "[An Activity Monitoring System to Support Classroom Research](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Montreal, Quebec, CANADA, pp. 1444-1449, June 2005.
43. Z. Xiang, H. Zhang, J. Huang, S. Song and K. Almeroth, "[A Hidden Environment Model for Constructing Indoor Radio Maps](#)," *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Taormina, ITALY, June 2005.
42. K. Harras, K. Almeroth and E. Belding, "[Delay Tolerant Mobile Networks \(DTMNs\): Controlled Flooding in Sparse Mobile Networks](#)," *IFIP Networking Conference*, Waterloo, Ontario, CANADA, May 2005.
41. A. Garyfalos and K. Almeroth, "[Coupons: Wide Scale Information Distribution for Wireless Ad Hoc Networks](#)," *IEEE Global Telecommunications Conference (Globecom) Global Internet and Next Generation Networks Symposium*, Dallas, Texas, USA, pp. 1655-1659, December 2004.
40. A. Knight and K. Almeroth, "[DeCAF: A Digital Classroom Application Framework](#)," *IASTED International Conference on Communications, Internet and Information Technology (CIIT)*, St. Thomas, US Virgin Islands, November 2004.
39. P. Namburi, K. Sarac and K. Almeroth, "[SSM-Ping: A Ping Utility for Source Specific Multicast](#)," *IASTED International Conference on Communications, Internet and Information Technology (CIIT)*, St. Thomas, US Virgin Islands, November 2004.
38. K. Ramachandran, E. Belding and K. Almeroth, "[DAMON: A Distributed Architecture for Monitoring Multi-hop Mobile Networks](#)," *IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, Santa Clara, California, USA, October 2004.
37. A. Garyfalos and K. Almeroth, "[Coupon Based Incentive Systems and the Implications of Equilibrium Theory](#)," *IEEE Conference on Electronic Commerce (CEC)*, San Diego, California, USA, pp. 213-220, July 2004.
36. A. Knight, K. Almeroth and B. Bimber, "[An Automated System for Plagiarism Detection Using the](#)



[Internet](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Lugano, Switzerland, pp. 3619-3625, June 2004.

35. H. Zhang and K. Almeroth, "[Supplement to Distance Learning: Design for a Remote TA Support System](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Lugano, Switzerland, pp. 2821-2830, June 2004.
34. U. Mohan, K. Almeroth and E. Belding, "[Scalable Service Discovery in Mobile Ad hoc Networks](#)," *IFIP Networking Conference*, Athens, Greece, pp. 137-149, May 2004.
33. V. Thanedar, K. Almeroth and E. Belding, "[A Lightweight Content Replication Scheme for Mobile Ad hoc Environments](#)," *IFIP Networking Conference*, Athens, Greece, pp. 125-136, May 2004.
32. R. Chalmers and K. Almeroth, "[A Mobility Gateway for Small-Device Networks](#)," *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, Orlando, Florida, USA, March 2004.
31. A. Jardosh, E. Belding, K. Almeroth and S. Suri, "[Towards Realistic Mobility Models For Mobile Ad hoc Networks](#)," *ACM Mobicom*, San Diego, California, USA, September 2003.
30. K. Sarac, P. Namburi and K. Almeroth, "[SSM Extensions: Network Layer Support for Multiple Senders in SSM](#)," *International Conference on Computer Communication and Networks (IC3N)*, Dallas, Texas, USA, October 2003.
29. K. Ramachandran and K. Almeroth, "[MAFIA: A Multicast Management Solution for Access Control and Traffic Filtering](#)," *IEEE/IFIP Conference on Management of Multimedia Networks and Services (MMNS)*, Belfast, Northern Ireland, September 2003.
28. J. Humfrey, S. Rollins, K. Almeroth, and B. Bimber, "[Managing Complexity in a Networked Learning Environment](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Honolulu, Hawaii, USA, pp. 60-63, June 2003.
27. K. Almeroth, S. Rollins, Z. Shen, and B. Bimber, "[Creating a Demarcation Point Between Content Production and Encoding in a Digital Classroom](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Honolulu, Hawaii, USA, pp. 2-5, June 2003.
26. M. Kolsch, K. Kvilekval, and K. Almeroth, "[Improving Speaker Training with Interactive Lectures](#)," *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Honolulu, Hawaii, USA, June 2003.
25. P. Rajvaidya and K. Almeroth, "[Analysis of Routing Characteristics in the Multicast Infrastructure](#)," *IEEE Infocom*, San Francisco, California, USA, April 2003.
24. S. Rollins and K. Almeroth, "[Pixie: A Jukebox Architecture to Support Efficient Peer Content Exchange](#)," *ACM Multimedia*, Juan Les Pins, FRANCE, December 2002.
23. S. Rollins, R. Chalmers, J. Blanquer, and K. Almeroth, "[The Active Information System\(AIS\): A Model for Developing Scalable Web Services](#)," *IASTED International Conference on Internet and Multimedia Systems and Applications (IMSA)*, Kauai, Hawaii, USA, August 2002.
22. S. Rollins and K. Almeroth, "[Seminal: Additive Semantic Content for Multimedia Streams](#)," *IASTED International Conference on Internet and Multimedia Systems and Applications (IMSA)*, Kauai, Hawaii, USA, August 2002.

21. B. Stiller, K. Almeroth, J. Altmann, L. McKnight, and M. Ott, "[Content Pricing in the Internet](#)," *SPIE ITCOM Conference on Internet Performance and Control of Network Systems (IPCNS)*, Boston, Massachusetts, USA, July 2002.
20. S. Jagannathan, J. Nayek, K. Almeroth and M. Hofmann, "[A Model for Discovering Customer Value for E-Content](#)," *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, Edmonton, Alberta, CANADA, July 2002.
19. S. Rollins and K. Almeroth, "[Deploying and Infrastructure for Technologically Enhanced Learning](#)," **Outstanding Paper** at the *World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA)*, Denver, Colorado, USA, pp. 1651-1656, June 2002.
18. P. Rajvaidya and K. Almeroth, "[Building the Case for Distributed Global Multicast Monitoring](#)," *Multimedia Computing and Networking (MMCN)*, San Jose, California, USA, January 2002.
17. S. Jagannathan and K. Almeroth, "[An Adaptive Pricing Scheme for Content Delivery Systems](#)," *IEEE Global Internet*, San Antonio, Texas, USA, November 2001.
16. K. Sarac and K. Almeroth, "[Providing Scalable Many-to-One Feedback in Multicast Reachability Monitoring Systems](#)," *IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS)*, Chicago, Illinois, USA, October 2001.
15. S. Jagannathan and K. Almeroth, "[The Dynamics of Price, Revenue and System Utilization](#)," *IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS)*, Chicago, Illinois, USA, October 2001.
14. A. Kanwar, K. Almeroth, S. Bhattacharyya, and M. Davy, "[Enabling End-User Network Monitoring via the Multicast Consolidated Proxy Monitor](#)," *SPIE ITCOM Conference on Scalability and Traffic Control in IP Networks (STCIPN)*, Denver, Colorado, USA, August 2001.
13. S. Jagannathan and K. Almeroth, "[Using Tree Topology for Multicast Congestion Control](#)," *International Conference on Parallel Processing (ICPP)*, Valencia, SPAIN, September 2001.
12. P. Rajvaidya and K. Almeroth, "[A Router-Based Technique for Monitoring the Next-Generation of Internet Multicast Protocols](#)," *International Conference on Parallel Processing (ICPP)*, Valencia, SPAIN, September 2001.
11. R. Chalmers and K. Almeroth, "[Modeling the Branching Characteristics and Efficiency Gains of Global Multicast Trees](#)," *IEEE Infocom*, Anchorage, Alaska, USA, April 2001.
10. R. Chalmers and K. Almeroth, "[Developing a Multicast Metric](#)," *Global Internet*, San Francisco, California, USA, December 2000.
9. K. Sarac and K. Almeroth, "[Monitoring Reachability in the Global Multicast Infrastructure](#)," *IEEE International Conference on Network Protocols (ICNP)*, Osaka, JAPAN, November 2000.
8. K. Almeroth, "[A Long-Term Analysis of Growth and Usage Patterns in the Multicast Backbone \(MBone\)](#)," *IEEE INFOCOM*, Tel Aviv, ISRAEL, March 2000.
7. K. Almeroth, K. Obraczka and D. De Lucia, "[A Lightweight Protocol for Interconnecting Heterogeneous Devices in Dynamic Environments](#)," *IEEE International Conference on Multimedia Computing and Systems (ICMCS)*, Florence, ITALY, June 1999.

6. K. Almeroth and M. Ammar, "[The Interactive Multimedia Jukebox \(IMJ\): A New Paradigm for the On-Demand Delivery of Audio/Video](#)," **Best Paper** at the *Seventh International World Wide Web Conference (WWW)*, Brisbane, AUSTRALIA, April 1998.
5. K. Almeroth, M. Ammar and Z. Fei, "[Scalable Delivery of Web Pages Using Cyclic Best-Effort \(UDP\) Multicast](#)," *IEEE INFOCOM*, San Francisco, California, USA, June 1998.
4. K. Almeroth and M. Ammar, "[Delivering Popular Web Pages Using Cyclic Unreliable Multicast \(Extended Abstract\)](#)," *SPIE Conference on Voice, Video and Data Communications*, Dallas, Texas, USA, November 1997.
3. K. Almeroth, A. Dan, D. Sitaram and W. Tetzlaff, "[Long Term Resource Allocation in Video Delivery Systems](#)," *IEEE INFOCOM*, Kobe, JAPAN, April 1997.
2. K. Almeroth and M. Ammar, "[On the Performance of a Multicast Delivery Video-On-Demand Service with Discontinuous VCR Actions](#)," *International Conference on Communications (ICC)*, Seattle, Washington, USA, June 1995.
1. K. Almeroth and M. Ammar, "[A Scalable, Interactive Video-On-Demand Service Using Multicast Communication](#)," *International Conference on Computer Communication and Networks (IC3N)*, San Francisco, California, USA, September 1994.

### C. Workshop Papers (refereed)

34. M. Tavakolifard, J. Gulla, K. Almeroth, F. Hopfgartner, B. Kille, T. Plumbaum, A. Lommatzsch, T. Brodt, A. Bucko, and T. Heintz, "[Workshop and Challenge on News Recommender Systems](#)," *ACM RecSys News Recommender Systems (NRS) Workshop and Challenge*, Hong Kong, CHINA, October 2013.
33. M. Tavakolifard, K. Almeroth, and J. Gulla, "[Does Social Contact Matter? Modelling the Hidden Web of Trust Underlying Twitter](#)," *ACM International Workshop on Social Recommender Systems (SRS)*, Rio de Janeiro, BRAZIL, May 2013.
32. D. Johnson, E. Belding, K. Almeroth and G. van Stam, "[Internet Usage and Performance Analysis of a Rural Wireless Network in Macha, Zambia](#)," *ACM Networked Systems for Developing Regions (NSDR) Workshop*, San Francisco, California, USA, June 2010.
31. D. Havey, R. Chertov, and K. Almeroth, "[Wired Wireless Broadcast Emulation](#)," *International Workshop on Wireless Network Measurement (WiNMe)*, Seoul, Korea, June 2009.
30. R. Raghavendra, P. Acharya, E. Belding, and K. Almeroth, "[MeshMon: A Multi-Tiered Framework for Wireless Mesh Network Monitoring](#)," *ACM Mobihoc Wireless of the Students, by the Students, for the Students Workshop (S3)*, New Orleans, Louisiana, USA, May 2009.
29. G. Swamynathan, C. Wilson, B. Boe, B. Zhao, and K. Almeroth, "[Do Social Networks Improve e-Commerce: A Study on Social Marketplaces](#)," *ACM Sigcomm Workshop on Online Social Networks (WOSN)*, Seattle, Washington, USA, August 2008.
28. R. Raghavendra, E. Belding, and K. Almeroth, "[Antler: A Multi-Tiered Approach to Automated Wireless Network Management](#)," *IEEE Workshop on Automated Network Management (ANM)*, Phoenix, Arizona, USA, April 2008.

27. S. Karpinski, E. Belding, and K. Almeroth, "[Towards Realistic Models of Wireless Workload](#)," *International Workshop on Wireless Network Measurement (WiNMee)*, Limassol, CYPRUS, April 2007.
26. K. Harras, M. Wittie, K. Almeroth, and E. Belding, "[ParaNets: A Parallel Network Architecture for Challenged Networks](#)," *IEEE Workshop on Mobile Computing Systems and Applications (HotMobile)*, Tucson, Arizona, USA, February 2007.
25. H. Caituiro-Monge, K. Almeroth, M. del Mar Alvarez-Rohena, "[Friend Relay: A Resource Sharing Framework for Mobile Wireless Devices](#)," *ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH)*, Los Angeles, California, September 2006.
24. G. Swamynathan, Ben Y. Zhao and K. Almeroth, "[Exploring the Feasibility of Proactive Reputations](#)," *International Workshop on Peer-to-Peer Systems (IPTPS)*, Santa Barbara, California, USA, February 2006.
23. G. Swamynathan, Ben Y. Zhao and K. Almeroth, "[Decoupling Service and Feedback Trust in a Peer-to-Peer Reputation System](#)," *International Workshop on Applications and Economics of Peer-to-Peer Systems (AEPP)*, Nanjing, CHINA, November 2005.
22. K. Ramachandran, M. Buddhikot, G. Chandranmenon, S. Miller, E. Belding, and K. Almeroth, "[On the Design and Implementation of Infrastructure Mesh Networks](#)," *IEEE Workshop on Wireless Mesh Networks (WiMesh)*, Santa Clara, California, USA, September 2005.
21. A. Jardosh, K. Ramachandran, K. Almeroth and E. Belding, "[Understanding Link-Layer Behavior in Highly Congested IEEE 802.11b Wireless Networks](#)," *Sigcomm Workshop on Experimental Approaches to Wireless Network Design and Analysis (EWIND)*, Philadelphia, Pennsylvania, USA, August 2005.
20. A. Sen Mazumder, K. Almeroth and K. Sarac, "[Facilitating Robust Multicast Group Management](#)," *Network and Operating System Support for Digital Audio and Video (NOSSDAV)*, Skamania, Washington, USA, June 2005.
19. Y. Sun, I. Sheriff, E. Belding and K. Almeroth, "[An Experimental Study of Multimedia Traffic Performance in Mesh Networks](#)," *MobiSys International Workshop on Wireless Traffic Measurements and Modeling (WitMeMo)*, Seattle, Washington, USA, June 2005.
18. K. Ramachandran, K. Almeroth and E. Belding, "[A Framework for the Management of Large-Scale Wireless Network Testbeds](#)," *International Workshop on Wireless Network Measurement (WiNMee)*, Trentino, ITALY, April 2005.
17. A. Garyfalos, K. Almeroth and K. Sanzgiri, "[Deployment Complexity Versus Performance Efficiency in Mobile Multicast](#)," *International Workshop on Broadband Wireless Multimedia: Algorithms, Architectures and Applications (BroadWiM)*, San Jose, California, USA, October 2004.
16. C. Ho, K. Ramachandran, K. Almeroth and E. Belding, "[A Scalable Framework for Wireless Network Monitoring](#)," *ACM International Workshop on Wireless Mobile Applications and Services on WLAN Hotspots (WMASH)*, Philadelphia, Pennsylvania, USA, October 2004.
15. A. Garyfalos, K. Almeroth and J. Finney, "[A Hybrid of Network and Application Layer Multicast for Mobile IPv6 Networks](#)," *International Workshop on Large-Scale Group Communication (LSGC)*, Florence, ITALY, October 2003.
14. A. Garyfalos, K. Almeroth and J. Finney, "[A Comparison of Network and Application Layer Multicast](#)



[for Mobile IPv6 Networks](#)," *ACM Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, San Diego, California, USA, September 2003.

13. S. Jagannathan, and K. Almeroth, "[Pricing and Resource Provisioning for Delivering E-Content On-Demand with Multiple Levels-of-Service](#)," *International Workshop on Internet Charging and QoS Technologies (ICQT)*, Zurich, SWITZERLAND, October 2002.
12. S. Rollins, K. Almeroth, D. Milojevic, and K. Nagaraja, "[Power-Aware Data Management for Small Devices](#)," *Workshop on Wireless Mobile Multimedia (WoWMoM)*, Atlanta, GA, USA, September 2002.
11. K. Almeroth, S. Bhattacharyya, and C. Diot, "[Challenges of Integrating ASM and SSM IP Multicast Protocol Architectures](#)," *International Workshop on Digital Communications: Evolutionary Trends of the Internet (IWDC)*, Taormina, ITALY, September 2001.
10. K. Sarac and K. Almeroth, "[Scalable Techniques for Discovering Multicast Tree Topology](#)," *Network and Operating System Support for Digital Audio and Video (NOSSDAV)* , Port Jefferson, New York, USA, June 2001.
9. P. Rajvaidya, K. Almeroth and K. Claffy, "[A Scalable Architecture for Monitoring and Visualizing Multicast Statistics](#)," *IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM)*, Austin, Texas, USA, December 2000.
8. S. Jagannathan, K. Almeroth and A. Acharya, "[Topology Sensitive Congestion Control for Real-Time Multicast](#)," *Network and Operating System Support for Digital Audio and Video (NOSSDAV)* , Chapel Hill, North Carolina, USA, June 2000.
7. K. Sarac and K. Almeroth, "[Supporting the Need for Inter-Domain Multicast Reachability](#)," *Network and Operating System Support for Digital Audio and Video (NOSSDAV)* , Chapel Hill , North Carolina, USA, June 2000.
6. D. Makofske and K. Almeroth, "[MHealth: A Real-Time Multicast Tree Visualization and Monitoring Tool](#)," *Network and Operating System Support for Digital Audio and Video (NOSSDAV)* , Basking Ridge New Jersey, USA, June 1999.
5. K. Almeroth and Y. Zhang, "[Using Satellite Links as Delivery Paths in the Multicast Backbone \(MBone\)](#)," *ACM/IEEE International Workshop on Satellite-Based Information Services (WOSBIS)*, Dallas, Texas, USA, October 1998.
4. M. Ammar, K. Almeroth, R. Clark and Z. Fei, "[Multicast Delivery of Web Pages OR How to Make Web Servers Pushy](#)," *Workshop on Internet Server Performance (WISP)*, Madison, Wisconsin, USA, June 1998.
3. K. Almeroth and M. Ammar, "[Prototyping the Interactive Multimedia Jukebox](#)," *Mini-conference on Multimedia Appliances, Interfaces, and Trials--International Conference on Communications (ICC)*, Montreal, Quebec, CANADA, June 1997.
2. K. Almeroth and M. Ammar, "[Collection and Modeling of the Join/Leave Behavior of Multicast Group Members in the MBone](#)," *High Performance Distributed Computing Focus Workshop (HPDC)*, Syracuse, New York, USA, August 1996.
1. K. Almeroth and M. Ammar, "[The Role of Multicast Communication in the Provision of Scalable and Interactive Video-On-Demand Service](#)," *Network and Operating System Support for Digital Audio and Video (NOSSDAV)* , Durham, New Hampshire, USA, April 1995.

## D. Non-Refereed Publications

8. K. Almeroth, E. Belding, M. Buddhikot, G. Chandranmenon, S. Miller, and K. Ramachandran, "[Infrastructure Mesh Networks](#)," *U.S. Patent Application US20070070959 A1*, September 2005.
7. K. Almeroth, R. Caceres, A. Clark, R. Cole, N. Duffield, T. Friedman, K. Hedayat, K. Sarac, M. Westerlund, "[RTP Control Protocol Extended Reports \(RTCP XR\)](#)," *Internet Engineering Task Force (IETF) Request for Comments (RFC) 3611*, November 2003.
6. Z. Albanna, K. Almeroth, D. Meyer, and M. Schipper, "[IANA Guidelines for IPv4 Multicast Address Allocation](#)," *Internet Engineering Task Force (IETF) Request for Comments (RFC) 3171*, August 2001.
5. B. Quinn and K. Almeroth, "[IP Multicast Applications: Challenges and Solutions](#)," *Internet Engineering Task Force (IETF), Request for Comments (RFC) 3170*, September 2001.
4. K. Almeroth, L. Wei and D. Farinacci, "[Multicast Reachability Monitor \(MRM\) Protocol](#)," *Internet Engineering Task Force Internet Draft*, July 2000.
3. K. Almeroth and L. Wei, "[Justification for and use of the Multicast Reachability Monitor \(MRM\) Protocol](#)," *Internet Engineering Task Force Internet Draft*, March 1999.
2. K. Almeroth, "[Managing IP Multicast Traffic: A First Look at the Issues, Tools, and Challenges](#)," IP Multicast Initiative White Paper, San Jose, California, USA, February 1999.
1. K. Almeroth, K. Obraczka and D. De Lucia, "[Pseudo-IP: Providing a Thin Network Protocol for Semi-Intelligent Wireless Devices](#)," *DARPA/NIST Smart Spaces Workshop*, Gaithersburg, Maryland, USA, July 1998.

## E. Released Software Systems

19. ***A Multi-radio Wireless Mesh Network Architecture*** -- <http://moment.cs.ucsb.edu/tic/>. Released December 1, 2006 (with K. Ramachandran, I. Sheriff, and E. Belding). The software as part of a multi-radio wireless mesh network that includes a Split Wireless Router that alleviates the interference that can occur between commodity radios within a single piece of hardware. The second is server software to perform channel assignment and communicate the assignments throughout the mesh network.
18. ***AODV-Spanning Tree (AODV-ST)*** -- <http://www.cs.ucsb.edu/~krishna/aodv-st/>. Released September 1, 2006 (with K. Ramachandran and E. Belding). AODV-ST is an extension of the well-known AODV protocol specifically designed for wireless mesh networks. The advantages of AODV-ST over AODV include support for high throughput routing metrics, automatic route maintenance for common-case traffic, and low route discovery latency.
17. ***The Multicast Detective*** -- [http://www.nmsl.cs.ucsb.edu/mcast\\_detective/](http://www.nmsl.cs.ucsb.edu/mcast_detective/). Released September 1, 2005 (with A. Sen Mazumder). The multicast detective is a robust solution to determine the existence and nature of multicast service for a particular user. By performing a series of tests, a user can determine whether there is network support for multicast, and consequently, whether a multicast group join is likely to succeed.

16. **AutoCap: Automatic and Accurate Captioning** -- <http://www.nmsl.cs.ucsb.edu/autocap/>. Released August 1, 2005 (with A. Knight). AutoCap is a software system that takes as input an audio/video file and a text transcript. AutoCap creates captions by aligning the utterances in the audio/video file to the transcript. For those words that are not recognized, AutoCap estimates when the words were spoken along with an error bound that gives the content creator an idea of caption accuracy. The result is a collection of accurately time-stamped captions that can be displayed with the video.
15. **PAIRwise Plagiarism Detection System** -- <http://cits.ucsb.edu/pair/>. Released July 1, 2005 (with A. Knight). PAIRwise is a plagiarism detection system with: (1) an easy-to-use interface for submitting papers, (2) a flexible comparison engine that allows intra-class, inter-class, and Internet-based comparisons, and (3) an intuitive graphical presentation of results.
14. **DAMON Multi-Hop Wireless Network Monitoring** -- <http://moment.cs.ucsb.edu/damon/>. Released October 1, 2004 (with K. Ramachandran and E. Belding). DAMON is a distributed system for monitoring multi-hop mobile networks. DAMON uses agents within the network to monitor network behavior and send collected measurements to data repositories. DAMON's generic architecture supports the monitoring of a wide range of protocol, device, or network parameters.
13. **Multicast Firewall** -- <http://www.nmsl.cs.ucsb.edu/mafia/>. Released June 1, 2004 (with K. Ramachandran). MAFIA, a multicast firewall and traffic management solution, has the specific aim of strengthening multicast security through multicast access control, multicast traffic filtering, and DoS attack prevention.
12. **AODV@IETF Peer Routing Software** -- <http://moment.cs.ucsb.edu/aodv-ietf/>. Released November 1, 2003 (with K. Ramachandran and E. Belding). One of the first large-scale efforts to run the Ad hoc On demand Distance Vector (AODV) routing protocol in a public space (at the Internet Engineering Task Force (IETF)). The implementation includes a daemon that runs on both the Linux and Windows operating systems.
11. **Mobility Obstacles** -- <http://moment.cs.ucsb.edu/mobility/>. Released September 1, 2003 (with A. Jardosh, E. Belding, and S. Suri). The topology and movement of nodes in ad hoc protocol simulation are key factors in protocol performance. In this project, we have developed ns-2 simulation plug-ins that create more realistic movement models through the incorporation of obstacles. These obstacles are utilized to restrict both node movement and wireless transmissions.
10. **mwalk** -- <http://www.nmsl.cs.ucsb.edu/mwalk/>. Released December 1, 2000 (with R. Chalmers). Mwalk is a collection of Java applications and Perl scripts which re-create a global view of a multicast session from mtrace and RTCP logs. Users to the site can download mwalk, examine the results of our analysis, or download data sets for use in simulations dependent on multicast tree characteristics.
9. **MANTRA2** -- <http://www.nmsl.cs.ucsb.edu/mantra/>. Released December 1, 1999 (with P. Rajvaidya). This new version of MANTRA focuses on the visualization of inter-domain routing statistics. Working in conjunction with the Cooperative Association for Internet Data Analysis (CAIDA) we have developed advanced collection and visualization techniques.
8. **MRM** -- <http://www.nmsl.cs.ucsb.edu/mrm/>. Released October 1, 1999 (with K. Sarac). MRM is the Multicast Reachability Protocol. We have implemented an end-host agent that responds to MRM Manager commands. Our end-host agent works in conjunction with Cisco routers to detect and isolate multicast faults.
7. **MANTRA** -- <http://www.nmsl.cs.ucsb.edu/mantra/>. Released January 1, 1999 (with P. Rajvaidya). MANTRA is the Monitoring and Analysis of Traffic in Multicast Routers. It uses scripts to collect and

display data from backbone multicast routers.

6. **SDR Monitor** -- <http://www.nmsl.cs.ucsb.edu/sdr-monitor/>. Released January 1, 1999 (with K. Sarac). The SDR Monitor receives e-mail updates from participants containing information about observed sessions in the MBone. A global view of multicast reachability is then constructed.
5. **The MHealth tool** -- <http://www.nmsl.cs.ucsb.edu/mhealth/>. Released September 1, 1998 (with D. Makofske). The mhealth tool graphically visualizes MBone multicast group trees and provides 'health' information including end-to-end losses per receiver and losses on a per hop basis. The implementation required expertise in Java, the MBone tools, and Unix.
4. **The MControl tool** -- <http://www.nmsl.cs.ucsb.edu/mcontrol/>. Released August 1, 1998 (with D. Makofske). Mcontrol is a tool to provide VCR-based interactivity for live MBone sessions. The implementation required expertise in Java, the MBone tools, and Unix.
3. **Interactive Multimedia Jukebox (IMJ)** -- <http://imj.ucsb.edu/>. Released October 1, 1996. The IMJ combines the WWW and the MBone conferencing tools to provide a multi-channel video jukebox offering both instructional and entertainment programming on a wide scale. The implementation required expertise in HTML, Perl, C, the MBone tools, and Unix.
2. **Mlisten** -- <http://www.cc.gatech.edu/computing/Telecomm/mbone/>. Released September 1, 1995. A tool to continuously collect MBone multicast group membership information including number and location of members, membership duration, and inter-arrival time for all audio and video sessions. The implementation required expertise in C, Tcl/Tk, the MBone tools, and UNIX socket programming.
1. **Audio-on-Demand (AoD)**. March 1, 1995. A server/client prototype to demonstrate interactivity in near VoD systems. The AoD server provides songs-on-demand and VCR-like functions via multicast IP over Ethernet. The implementation required expertise in C, OpenWindows programming, UNIX socket programming, and network programming.

## F. Tutorials, Panels and Invited Talks

- "25th Anniversary Panel," Network and Operating System Support for Digital Audio and Video (NOSSDAV), Portland, Oregon, USA, March 2015.
- "Sensing and Opportunistic Delivery of Ubiquitous Video in Health Monitoring, On-Campus and Social Network Applications," Workshop on Mobile Video Delivery (MoViD), Chapel Hill North Carolina, USA, February 2012.
- "Medium Access in New Contexts: Reinventing the Wheel?," USC Invited Workshop on Theory and Practice in Wireless Networks, Los Angeles, California, USA, May 2008.
- "The Wild, Wild West: Wireless Networks Need a New Sheriff," University of Florida CISE Department Lecture Series, Gainesville, Florida, USA, February 2008.
- "Distinguishing Between Connectivity, Intermittent Connectivity, and Intermittent Disconnectivity," Keynote at the ACM MobiCom Workshop on Challenged Networks (CHANTS), Montreal, CANADA, September 2007.
- "The Three Ghosts of Multicast: Past, Present, and Future," Keynote at the Trans-European Research and Education Networking Association (TERENA) Networking Conference, Lynby, DENMARK, May



2007.

- "Multicast Help Wanted: From Where and How Much?," Keynote at the Workshop on Peer-to-Peer Multicasting (P2PM), Las Vegas, Nevada, USA, January 2007.
- "The Confluence of Wi-Fi and Apps: What to Expect Next," Engineering Insights, UC-Santa Barbara, Santa Barbara, California, USA, October 2006.
- "Challenges, Opportunities, and Implications for the Future Internet," University of Minnesota Digital Technology Center, Minneapolis, Minnesota, USA, September 2006.
- "Wireless Technology as a Catalyst: Possibilities for Next-Generation Interaction," Santa Barbara Forum on Digital Transitions, Santa Barbara, California, USA, April 2006.
- "Challenges and Opportunities in an Internet with Pervasive Wireless Access," University of Texas--Dallas Computer Science Colloquium, Dallas, Texas, USA, March 2006.
- "Challenges and Opportunities with Pervasive Wireless in the Internet," Duke University Computer Science Colloquium, Durham, North Carolina, USA, February 2006.
- "The Span From Wireless Protocols to Social Applications," Intel Research Labs, Cambridge, United Kingdom, December 2005.
- "The Internet Dot.Com Bomb and Beyond the Dot.Com Calm," CSE IGERT and Cal Poly Lecture Series, San Luis Obispo, California, USA, October 2005.
- "Panel: Directions in Networking Research," IEEE Computer Communications Workshop (CCW), Irvine, California, USA, October 2005.
- "Economic Incentives for Ad Hoc Networks," KAIST New Applications Seminar, Seoul, South Korea, March 2004.
- "New Applications for the Next Generation Internet," Citrix Systems, Santa Barbara, California, USA, March 2004.
- "PI: The Imperfect Pursuit of Pure Pattern," CITS Visions in Technology Series, Santa Barbara, California, USA, January 2004.
- "Panel: Core Networking Issues and Protocols for the Internet," National Science Foundation (NSF) Division of Advanced Networking Infrastructure and Research (ANIR) Principal Investigators Workshop, Washington DC, USA, March 2003.
- "Panel: Pricing for Content in the Internet," SPIE ITCom Internet Performance and Control of Network Systems, Boston, Massachusetts, USA, July 2002.
- "The Technology Behind Wireless LANs," Central Coast MIT Enterprise Forum, Santa Barbara, California, USA, March 2002.
- "Lessons Learned in the Digital Classroom," Center for Information and Technology Brown Bag Symposium, Santa Barbara, California, USA, March 2002.
- "The Evolution of Advanced Networking Services: From the ARPAnet to Internet2," California State University--San Luis Obispo CS Centennial Colloquium Series, San Luis Obispo, California, USA, February 2002.

- "Deployment of IP Multicast in Campus Infrastructures," Internet2 Campus Deployment Workshop, Atlanta, Georgia, USA, May 2001.
- "Multicast: Is There Anything Else to Do?," Sprint Research Retreat, Miami, Florida, USA, May 2001.
- "The Evolution of Next-Generation Internet Services and Applications," Government Technology Conference 2001 (GTC) for the Western Region, Sacramento, California, USA, May 2001.
- "I2 Multicast: Not WIDE-scale Deployment, FULL-scale Deployment," Closing Plenary, Internet2 Member Meetings, Washington, D.C., USA, March 2001.
- "Panel: Beyond IP Multicast," Content Delivery Networks (CDN), New York, New York, USA, February 2001.
- "Viable Multicast Pricing & Business Models for Wider-Scale Deployment," Content Delivery Networks (CDN), New York, New York, USA, February 2001.
- "IP Multicast: Modern Protocols, Deployment, and Management," Content Delivery Networks (CDN), New York, New York, USA, February 2001 & San Jose, California, USA, December 2001.
- "Under the Hood of the Internet," Technology 101: Technology for Investors, Center for Entrepreneurship & Engineering Management, November 2000.
- "Understanding Multicast Traffic in the Internet," (1) University of Virginia, (2) University of Maryland, and (3) Columbia University, September 2000.
- "The Bad, The Ugly, and The Good: The Past, Present, and Future of Multicast," Digital Fountain, San Francisco, California, USA, August 2000.
- "Implications of Source-Specific Multicast (SSM) on the Future of Internet Content Delivery," Occam Networks, Santa Barbara, California, USA, August 2000.
- "Introduction to Multicast Routing Protocols," UC-Berkeley Open Mash Multicast Workshop, Berkeley, California, USA, July 2000.
- "Efforts to Understand Traffic and Tree Characteristics," University of Massachusetts--Amherst Colloquia, Amherst, Massachusetts, USA, May 2000.
- "Monitoring Multicast Traffic," Sprint Research Retreat, Half Moon Bay, California, USA, April 2000.
- "What is the Next Generation of Multicast in the Internet?," HRL Laboratories, Malibu, California, USA, January 2000.
- "Mission and Status of the Center for Information Technology and Society (CITS)," Intel Research Council, Portland, Oregon, USA, September 1999.
- "Multicast at a Crossroads," IP Multicast Initiative Summits and Bandwidth Management Workshops, San Francisco, CA, USA, (1) October 1999; (2) February 2000; and (3) June 2000.
- "IP Multicast: Modern Protocols, Deployment, and Management," Networld+Interop: (1) Las Vegas, Nevada, USA--May 2000; (2) Tokyo, JAPAN--June 2000; (3) Atlanta, Georgia, USA--September 2000; (4) Las Vegas, Nevada, USA--May 2001; (5) Las Vegas, Nevada, USA--May 2002.
- "IP Multicast: Practice and Theory" (w/ Steve Deering), Networld+Interop: (1) Las Vegas, Nevada,

USA--May 1999; (2) Tokyo, JAPAN--June 1999; and (3) Atlanta, Georgia, USA--September 1999.

- "Internet2 Multicast Testbeds and Applications," Workshop on Protocols for High Speed Networks (PfHSN), Salem, Massachusetts, USA, August 1999.
- "IP Multicast: Protocols for the Intra- and Inter-Domain," Lucent Technologies, Westford, Massachusetts, USA, August 1999.
- "Internet2 Multicast Testbeds and Applications," NASA Workshop: Bridging the Gap, Moffett Field, California, USA, August 1999.
- "The Evolution of Next-Generation Services and Applications in the Internet," Tektronix Distinguished Lecture Series, Portland, Oregon, USA, May 1999.
- "Multicast Applications and Infrastructure in the Next Generation Internet," CENIC 99 Workshop on Achieving Critical Mass for Advanced Applications, Monterey, California, USA, May 1999.
- "Multicast Traffic Monitoring and Analysis Work at UCSB" (w/ P. Rajvaidya), Workshop on Internet Statistics and Metrics Analysis (ISMA), San Diego, California, USA, April 1999.
- "How the Internet Works: Following Bits Around the World," Science Lite, Santa Barbara General Affiliates and Office of Community Relations, California, USA, February 1999.
- "Managing Multicast: Challenges, Tools, and the Future," IP Multicast Initiative Summit, San Jose, California, USA, February 1999.
- "The Future of Multicast Communication and Protocols," Internet Bandwidth Management Summit (iBAND), San Jose, California, USA, November 1998.
- "An Overview of IP Multicast: Applications and Deployment," (1) Workshop on Evaluating IP Multicast as the Solution for Webcasting Real-Time Networked Multimedia Applications, New York, New York, USA, July 1998; and (2) Satellites and the Internet Conference, Washington, D.C., USA, July 1998.
- "IETF Developments in IP Multicast," IP Multicast Initiative Summit, San Jose, California, USA, February 1998.
- "An Introduction to IP Multicast and the Multicast Backbone (MBone)" vBNS Technical Meeting sponsored by the National Center for Network Engineering (NLNR), San Diego, California, USA, February 1998.
- "Using Multicast Communication to Deliver WWW Pages" Computer Communications Workshop (CCW '97), Phoenix, Arizona, USA, September 1997.

## G. Research Funding

- K. Almeroth, "Packet Scheduling Using IP Embedded Transport Instrumentation," Cisco Systems Inc., \$100,000, 3/1/13-8/31/14.
- K. Almeroth, E. Belding and S.J. Lee, "GOALI: Maximizing Available Bandwidth in Next Generation WLANs", National Science Foundation (NSF), \$101,088, 10/1/13-9/30/14.

- K. Almeroth and E. Belding, "GOALI: Intelligent Channel Management in 802.11n Networks," National Science Foundation (NSF), \$51,000, 10/1/10-9/30/11.
- B. Zhao, K. Almeroth, H. Zheng, and E. Belding, "NeTS: Medium: Airlab: Distributed Infrastructure for Wireless Measurements," National Science Foundation (NSF), \$700,000, 9/1/09-8/13/13.
- K. Almeroth, E. Belding and T. Hollerer, "NeTS-WN: Wireless Network Health: Real-Time Diagnosis, Adaptation, and Management," National Science Foundation (NSF), \$600,000, 10/1/07-9/30/10.
- K. Almeroth, "Next-Generation Service Engineering in Internet2," University Consortium for Advanced Internet Development (UCAID), \$1,254,000, 7/1/04-6/30/09 (reviewed and renewed each year).
- B. Manjunath, K. Almeroth, F. Bullo, J. Hespanha, T. Hollerer, C. Krintz, U. Madhow, K. Rose, A. Singh, and M. Turk, "Large-Scale Multimodal Wireless Sensor Network," Office of Naval Research Defense University Research Instrumentation Program (DURIP), \$655,174, 4/14/08-4/14/09.
- K. Almeroth and E. Belding, "Improving Robustness in Evolving Wireless Infrastructures," Intel Corporation, \$135,000, 7/1/06-6/30/09 (reviewed and renewed for second and third year).
- K. Almeroth and K. Sarac, "Bridging Support in Mixed Deployment Multicast Environments," Cisco Systems Inc., \$100,000, 9/1/07-8/31/08.
- K. Sarac and K. Almeroth, "Building the Final Piece in One-to-Many Content Distribution," Cisco Systems Inc., \$95,000, 9/1/06-8/31/07.
- E. Belding, K. Almeroth and J. Gibson, "Real-Time Communication Support in a Ubiquitous Next-Generation Internet," National Science Foundation (NSF), \$900,000, 10/1/04-9/30/07.
- K. Almeroth and K. Sarac, "Improving the Robustness of Multicast in the Internet," Cisco Systems Inc., \$80,000, 9/1/04-8/31/05.
- R. Mayer, B. Bimber, K. Almeroth and D. Chun, "Assessing the Pedagogical Implications of Technology in College Courses," Mellon Foundation, \$350,000, 7/1/04-6/30/07.
- B. Bimber, A. Flanagan and C. Stol, "Technological Change and Collective Association: Changing Relationships Among Technology, Organizations, Society and the Citizenry," National Science Foundation (NSF), \$329,175, 7/1/04-6/30/07.
- K. Almeroth and B. Bimber, "Plagiarism Detection Techniques and Software," UCSB Instructional Development, \$22,000, 7/1/04-6/30/05.
- K. Almeroth, "Student Travel Support for the 14th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)," National Science Foundation (NSF), \$10,000, 5/1/04-8/31/04.
- K. Almeroth, "An Automated Indexing System for Remote, Archived Presentations," QAD Inc., \$25,000, 5/1/04-6/30/05.
- K. Almeroth and M. Turk, "A Remote Teaching Assistant Support System," Microsoft, \$40,000, 1/1/04-6/30/05.
- K. Almeroth, "Supporting Multicast Service Functionality in Helix," Real Networks, \$30,000, 9/1/03-6

/30/04.

- K. Almeroth and E. Belding, "Service Discovery in Mobile Networks," Nokia Summer Research Grant (U. Mohan), \$10,240, 7/1/03-9/30/03.
- K. Almeroth, D. Zappala, "Building a Global Multicast Service," Cisco Systems Inc., \$100,000, 1/1/03-6/30/04.
- K. Almeroth, "Developing A Dynamic Protocol for Candidate Access Router Discovery," Nokia Graduate Student Fellowship (R. Chalmers), \$26,110, 9/01/02-6/30/03.
- B. Bimber and K. Almeroth, "The Role of Collaborative Groupware in Organizations," Toole Family Foundation, \$182,500 (\$20,000 cash plus \$162,500 in software), 9/1/02-8/30/07.
- B. Manjunath, et al., "Digital Multimedia: Graduate Training Program in Interactive Digital Multimedia," National Science Foundation (NSF), \$2,629,373, 4/1/02-3/31/07.
- J. Green, K. Almeroth, et al., "Inquiry in the Online Context: Learning from the Past, Informing the Future," UCSB Research Across Disciplines, \$10,000, 9/1/01-8/31/02.
- K. Almeroth, "Monitoring and Maintaining the Global Multicast Infrastructure," Cisco Systems Inc., \$54,600, 7/1/01-6/30/02.
- R. Kemmerer, K. Almeroth, et al., "Hi-DRA High-speed, Wide-area Network Detection, Response, and Analysis," Department of Defense (DoD), \$4,283,500, 5/1/01-4/30/06.
- A. Singh, K. Almeroth, et al., "Digital Campus: Scalable Information Services on a Campus-wide Wireless Network," National Science Foundation (NSF), 1,450,000, 9/15/00-12/31/04.
- K. Almeroth, "Visualizing the Global Multicast Infrastructure," UC MICRO w/ Cisco Systems Inc., \$85,438, 7/1/00-6/30/02.
- H. Lee, K. Almeroth, et al., "Dynamic Sensing Systems," International Telemetering Foundation, \$260,000, 07/01/00-06/30/04.
- B. Bimber and K. Almeroth, "Funding for the Center on Information Technology and Society," \$250,000 from Dialogic (an Intel Company) and \$250,000 from Canadian Pacific.
- K. Almeroth, "CAREER: From Protocol Support to Applications: Elevating Multicast to a Ubiquitous Network Service," National Science Foundation (NSF), \$200,000, 9/1/00-8/31/04.
- K. Almeroth, "Characterizing Multicast Use and Efficiency in the Inter-Domain," Sprint Advanced Technology Laboratories, \$62,500, 3/1/00-6/30/01.
- K. Almeroth, "Producing the Next Generation of Multicast Monitoring and Management Protocols and Tools," UC MICRO w/ Cisco Systems Inc., \$124,500, 7/1/99 - 6/30/01.
- K. Almeroth, "Utilizing Satellite Links in the Provision of an Inter-Wide Multicast Service," HRL Laboratories, \$20,000, 7/1/99 - 6/30/00.
- T. Smith, K. Almeroth, et al., "Alexandria Digital Earth Prototype," National Science Foundation, \$5,400,000, 4/1/99-3/31/04.
- V. Vesna, K. Almeroth, et al., "Online Public Spaces: Multidisciplinary Explorations in Multi-User

Environments (OPS:MEME), Phase II," UCSB Research Across Disciplines, \$50,000, 9/1/98-8/31/99.

- K. Almeroth, "Techniques and Analysis for the Provision of Multicast Route Management," UC MICRO w/ Cisco Systems Inc., \$97,610, 7/1/98 - 6/30/00.
- K. Almeroth, "Capturing and Modeling Multicast Group Membership in the Multicast Backbone (MBone)," UC MICRO w/ Hughes Research Labs, \$19,146, 7/1/98 - 12/31/99.
- K. Almeroth, "Building a Content Server for the Next Generation Digital Classroom," UCSB Faculty Research Grant, \$5,000, 7/1/98-6/31/99.

## **H. Research Honors and Awards**

- IEEE Fellow Status, 2013
- Finalist for Best Paper Award, IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON), June 2008
- Best Paper Award, Passive and Active Measurement (PAM) Conference, April 2007
- Outstanding Paper Award, World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA), June 2006
- IEEE Senior Member Status, 2003
- Finalist for Best Student Paper Award, ACM Multimedia, December 2002
- Outstanding Paper Award, World Conference on Educational Multimedia, Hypermedia & Telecommunications (ED MEDIA), June 2002
- Computing Research Association (CRA) Digital Government Fellowship, 2001
- National Science Foundation CAREER Award, 2000
- Best Paper Award, 7th International World Wide Web Conference, April 1998

## **III. Service**

### **A. Professional Activities**

#### **1. Society Memberships**

Member, Association for Computing Machinery (ACM): 1993-present  
Member, ACM Special Interest Group on Communications (SIGComm): 1993-present  
Fellow, Institute of Electrical and Electronics Engineers (IEEE): 1993-present  
Member, IEEE Communications Society (IEEE ComSoc): 1993-present  
Member, American Society for Engineering Education (ASEE): 2003-2006

#### **2. Review Work for Technical Journals and Publishers**

NSF CISE research proposals, IEEE/ACM Transactions on Networking, IEEE/ACM Transactions on Computers, IEEE/ACM Transactions on Communications, IEEE Transactions on



Circuits and Systems for Video Technology, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Multimedia, IEEE Communications, IEEE Communications Letters, IEEE Network, IEEE Internet Computing, IEEE Multimedia, IEEE Aerospace & Electronics Systems Magazine, ACM Transactions on Internet Technology, ACM Transactions on Multimedia Computing, Communications and Applications, ACM Computing Surveys, ACM Computer Communications Review, ACM Computeres in Entertainment, ACM/Springer Multimedia Systems Journal, AACE Journal of Interactive Learning (JILR), International Journal of Computer Mathematics, Journal of Communications and Networks, Journal of Parallel and Distributed Computing, Journal of Network and Systems Management, Journal of High Speed Networking, Journal of Communications and Networks, Journal on Selected Areas in Communications, Journal of Wireless Personal Communications, Personal Mobile Communications, Annals of Telecommunications, International Journal of Wireless and Mobile Computing, Pervasive and Mobile Computing (PMC), Wireless Networks Journal, Computer Networks Journal, Cluster Computing, Computer Communications, Mobile Computing and Communications Review, Performance Evaluation, Software--Practice & Experience, Information Processing Letters, ACM Sigcomm, ACM Multimedia, ACM Network and System Support for Digital Audio and Video Workshop (NOSSDAV), ACM Sigcomm Workshop on the Economics of Peer-to-Peer Systems (P2PEcon), ACM Sigcomm Workshop on Challenged Networks (CHANTS), IEEE Infocom, IEEE Globecom, IEEE Global Internet (GI) Symposium, IEEE Globecom Automatic Internet Symposium, IEEE Globecom Internet Services and Enabling Technologies (IS&ET) Symposium, IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), IEEE International Conference on Network Protocols (ICNP), IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON), IEEE International Conference on Multimedia and Exposition (ICME), IEEE International Conference on Communications (ICC), IEEE International Conference on Parallel and Distributed Systems (ICPADS) IEEE International Symposium on High-Performance Distributed Computing (HPDC), IEEE International Conference on Distributed Computing Systems (ICDCS), IEEE International Workshop on Quality of Service (IWQoS), IEEE/IFIP Network Operations and Management Symposium (NOMS), IFIP/IEEE International Symposium on Integrated Network Management (IM), IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS), IEEE Aerospace & Electronics Systems Magazine, SPIE Conference on Multimedia Computing and Networking (MMCN), IFIP Networking, IASTED International Conference on Information Systems and Databases (ISD), IASTED International Conference on Communications, Internet, and Information Technology, IASTED International Conference on Internet and Multimedia Systems and Applications (IMSA), IASTED International Conference on European Internet and Multimedia Systems and Applications (EuroIMSA), IASTED International Conference on Communications and Computer Networks (CCN), IASTED International Conference on Software Engineering and Applications (SEA), International Conference on Computer and Information Science (ICIS), International Association for Development of the Information Society (IADIS) International Conference on the WWW/Internet, Workshop on Network Group Communication (NGC), International Conference on Next Generation Communication (CoNEXT), International Conference on Parallel Processing (ICPP), International Conference on Computer Communications and Networks (IC3N), International Workshop on Hot Topics in Peer-to-Peer Systems (Hot-P2P), International Workshop on Wireless Network Measurements (WiNMee), International Workshop on Incentive-Based Computing (IBC), International Workshop on Multi-hop Ad Hoc Networks (REALMAN), International Workshop on Broadband Wireless Multimedia: Algorithms, Architectures and Applications (BroadWIM), International Packet Video (PV) Workshop, High Performance Networking Conference (HPN), International Parallel Processing Symposium (IPPS), International Symposium on Innovation in Information & Communication Technology (ISIICT), Workshop on Coordinated Quality of Service in Distributed Systems (COQODS), Pearson

Education (Cisco Press) Publishers, Macmillan Technical Publishing, and Prentice Hall Publishers.

### **3. Conference Committee Activities**

#### **Journal/Magazine Editorial Board**

IEEE Transactions on Mobile Computing (TMC): 2006-2011, 2017-present (Associate Editor-in-Chief)  
IEEE Networking Letters: 2018-present  
IEEE Transactions on Network and Service Management (TNSM): 2015-present  
Journal of Network and Systems Management (JNSM): 2011-present  
IEEE/ACM Transactions on Networking (ToN): 2003-2009, 2013-2017  
ACM Computers in Entertainment: 2002-2015  
IEEE Network: 1999-2012  
AACE Journal of Interactive Learning Research (JILR): 2003-2012  
IEEE Transactions on Mobile Computing (TMC): 2006-2011  
ACM Computer Communications Review (CCR): 2006-2010

#### **Journal/Magazine Guest Editorship**

IEEE Journal on Selected Areas in Communications (JSAC) Special Issue on "Delay and Disruption Tolerant Wireless Communication", June 2008  
Computer Communications Special Issue on "Monitoring and Measuring IP Networks", Summer 2005  
Computer Communications Special Issue on "Integrating Multicast into the Internet", March 2001

#### **Conference/Workshop Steering Committee**

IEEE International Conference on Network Protocols (ICNP): 2007-present  
ACM Sigcomm Workshop on Challenged Networks (CHANTS): 2006-present  
International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV): 2001-present, 2005-2011 (chair), 2012-present (co-chair)  
IEEE Global Internet (GI) Symposium: 2005-2013, 2018-present  
IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS): 2005-2009

#### **Conference/Workshop Chair**

International Conference on Communication Systems and Networks (COMSNETS): 2014 (co-chair)  
ACM International Conference on Next Generation Communication (CoNext): 2013 (co-chair)  
ACM RecSys News Recommender Systems (NRS) Workshop and Challenge: 2013 (co-chair)  
ACM Sigcomm Workshop on Challenged Networks (CHANTS): 2006 (co-chair)  
IEEE International Conference on Network Protocols (ICNP): 2003 (co-chair), 2006  
International Workshop on Wireless Network Measurements (WiNMee): 2006 (co-chair)  
IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS): 2002 (co-chair)  
International Workshop on Network and Operating System Support for Digital Audio and



Video (NOSSDAV): 2002 (co-chair), 2003 (co-chair)  
IEEE Global Internet (GI) Symposium: 2001 (co-chair), 2018 (co-chair)  
International Workshop on Networked Group Communication (NGC): 2000 (co-chair)

### **Program Chair**

International Conference on Computer Communication and Networks (ICCCN): 2015  
(Track co-chair) International Conference on Communication Systems and Networks  
(COMSNETS): 2010  
IEEE International Conference on Network Protocols (ICNP): 2008 (co-chair)  
IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON): 2007  
(co-chair)  
IFIP Networking: 2005 (co-chair)

### **Posters/Demonstrations Chair**

ACM Sigcomm: 2012 (co-chair)

### **Student Travel Grants Chair**

ACM Sigcomm: 2010 (co-chair)

### **Publicity Chair**

IFIP/IEEE International Conference on Management of Multimedia Networks and Services  
(MMNS): 2004 (co-chair)

### **Keynote Chair**

IEEE Infocom: 2005 (co-chair)

### **Local Arrangements Chair**

Internet2 "Field of Dreams" Workshop: 2000

### **Tutorial Chair**

ACM Multimedia: 2000  
IEEE International Conference on Network Protocols (ICNP): 1999

### **Panel/Session Organizer**

NSF ANIR PI 2003 Panel on "Core Networking Issues and Protocols for the Internet"  
CCW 2001 Session on "Multicast/Peer-to-Peer Networking"  
NOSSDAV 2001 Panel on "Multimedia After a Decade of Research"  
NGC 2000 Panel on "Multicast Pricing"

### **Technical Program Committee**

IEEE International Conference on Network Protocols (ICNP): 1999, 2000, 2001, 2003,  
2004, 2005, 2006, 2007, 2008, 2009 (Area Chair), 2010 (Area Chair), 2011 (Area Chair),  
2012 (Area Chair), 2013, 2014 (Area Chair), 2015 (Area Chair), 2016 (Area Chair), 2017  
(Area Chair), 2018 (Area Chair), 2019 (Area Chair)  
International Workshop on Network and Operating System Support for Digital Audio and

Video (NOSSDAV): 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019

ACM Multimedia (MM): 2001, 2003, 2004, 2005 (short paper), 2006, 2007, 2008, 2008 (short paper), 2010, 2011, 2012, 2013, 2015, 2016, 2017, 2018, 2019

IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON): 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011 (Area Chair), 2012 (Area Chair), 2013, 2014 (Area Chair), 2015, 2016 (Area Chair), 2017, 2018, 2019

IEEE/IFIP Network Operations and Management Symposium (NOMS): 2004, 2006, 2010

IEEE Infocom: 2004, 2005, 2006, 2008, 2009, 2010 (Area Chair), 2011 (Area Chair), 2012 (Area Chair)

IFIP Networking: 2004, 2005, 2006, 2007, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018

IEEE International Conference on Communications (ICC) Next Generation Networking and Internet Symposium (NGNI): 2018, 2019

ACM Workshop on Mobile Video (MoVid): 2014, 2015, 2016, 2017

ACM Student Research Competition (SRC) Grand Finals: 2014

Mobile and Social Computing for Collaborative Interactions (MSC): 2014

IEEE Conference on Communications and Network Security (CNS): 2013

IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM): 2005, 2006, 2007, 2008, 2009, 2010

ACM Sigcomm Workshop on Challenged Networks (CHANTS): 2006, 2008, 2009, 2010, 2011, 2012, 2016, 2017, 2018, 2019

IEEE International Conference on Distributed Computing Systems (ICDCS): 2006, 2010, 2011, 2012, 2013

International Workshop on Wireless Network Measurements (WiNMee): 2006, 2008, 2010

ACM Sigcomm: 2008 (poster), 2010

IEEE International Conference on Computer Communication and Networks (IC3N): 2008, 2009, 2010, 2011, 2012

International Conference on Communication Systems and Networks (COMSNETS): 2009, 2010, 2011, 2012, 2013

International Conference on Sensor Networks (SENSORNETS): 2012

International Workshop on Social and Mobile Computing for Collaborative Environments (SOMOCO): 2012

Workshop on Scenarios for Network Evaluation Studies (SCENES): 2009, 2010, 2011

ACM Multimedia Systems (MMSys): 2010, 2011, 2012, 2015, 2016, 2017, 2018, 2019

IEEE International Symposium on Multimedia (ISM): 2016

IEEE International Conference on Pervasive Computing and Communications (PerCom): 2010

IEEE Wireless Communications and Networking Conference (WCNC): 2010, 2011

ACM International Symposium on Mobility Management and Wireless Access (MobiWac): 2010, 2011

International Conference on Computing, Networking and Communications, Internet Services and Applications Symposium (ICNC-ISA): 2012, 2013

IEEE WoWMoM Workshop on Hot Topics in Mesh Networking (HotMesh): 2010, 2011, 2012, 2013

IEEE Workshop on Pervasive Group Communication (PerGroup): 2010

ACM International Conference on Next Generation Communication (CoNEXT): 2005, 2006, 2007, 2009, 2012

IEEE International Conference on Broadband Communications, Networks, and Systems (BroadNets) Wireless Communications, Networks and Systems Symposium: 2007, 2008, 2009

IEEE International Conference on Broadband Communications, Networks, and Systems (BroadNets) Internet Technologies Symposium: 2007, 2008, 2009  
 International Workshop on Mobile and Networking Technologies for Social Applications (MONET): 2008, 2009  
 Extreme Workshop on Communication-The Midnight Sun Expedition (ExtremeCom): 2009  
 IEEE International Workshop on Cooperation in Pervasive Environments (CoPE): 2009  
 International Workshop on the Network of the Future (FutureNet): 2009, 2010, 2011, 2012  
 IEEE International Conference on Multimedia and Exposition (ICME): 2010  
 SPIE Conference on Multimedia Computing and Networking (MMCN): 2004, 2008  
 ACM Sigcomm Workshop on the Economics of Networks, Systems, and Computation (NetEcon): 2008  
 IEEE International Conference on Communications (ICC): 2008  
 IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS): 2008  
 IFIP/IEEE International Symposium on Integrated Network Management (IM): 2005, 2007  
 Global Internet (GI) Symposium: 2001, 2002, 2004, 2006, 2007  
 IEEE/ACM International Conference on High Performance Computing (HiPC): 2007  
 ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc): 2007  
 IEEE Workshop on Embedded Systems for Real-Time Multimedia (ESTIMedia): 2007  
 IEEE/IFIP Wireless On Demand Network Systems and Services (WONS): 2007  
 IFIP/IEEE International Conference on Management of Multimedia Networks and Services (MMNS): 2001, 2002, 2003, 2004, 2005, 2006  
 IASTED International Conference on European Internet and Multimedia Systems and Applications (EuroIMSA): 2004, 2006  
 IEEE International Conference on Parallel and Distributed Systems (ICPADS): 2005, 2006  
 IEEE Globecom Internet Services and Enabling Technologies (IS&ET) Symposium: 2006  
 International Workshop on Incentive-Based Computing (IBC): 2006  
 IEEE International Workshop on Quality of Service (IWQoS): 2006, 2014, 2015  
 International Workshop on Multi-hop Ad Hoc Networks (REALMAN): 2006  
 IEEE Globecom Automatic Internet Symposium: 2005  
 ACM Sigcomm Workshop on the Economics of Peer-to-Peer Systems (P2PEcon): 2005  
 International Conference on Parallel Processing (ICPP): 2001, 2003, 2004  
 International Packet Video (PV) Workshop: 2002, 2003, 2004  
 IEEE International Symposium on High-Performance Distributed Computing (HPDC): 2004  
 ACM Sigcomm: 2004 (poster)  
 International Workshop on Broadband Wireless Multimedia: Algorithms, Architectures and Applications (BroadWIM): 2004  
 International Symposium on Innovation in Information & Communication Technology (ISIICT): 2004  
 Workshop on Coordinated Quality of Service in Distributed Systems (COQODS): 2004  
 IASTED International Conference on Networks and Communication Systems (NCS): 2004  
 IASTED International Conference on Communications, Internet, and Information Technology (CIIT): 2004  
 IASTED International Conference on Internet and Multimedia Systems and Applications (IMSA): 2003, 2004  
 International Workshop on Networked Group Communication (NGC): 1999, 2000, 2001, 2002, 2003  
 International Association for Development of the Information Society (IADIS)  
 International Conference on the WWW/Internet: 2003

International Conference on Computer and Information Science (ICIS): 2003  
Human.Society@Internet: 2003  
IASTED International Conference on Communications and Computer Networks (CCN): 2002  
The Content Delivery Networks (CDN) Event: 2001  
IP Multicast Initiative Summit: 1998, 1999, 2000  
Corporation for Education Network Initiatives in California (CENIC): 1999  
Internet Bandwidth Management Summit (iBAND): 1998, 1999

## **B. Technical Activities**

### **1. Working Groups**

Internet2 Working Group on Multicast, Chair: 1998-2005  
IEEE Communications Society Internet Technical Committee (ITC), Conference Coordinator: 2000-2004  
IETF Multicast Directorate (MADDOGS), Member: 1999-2001  
IASTED Technical Committee on the Web, Internet and Multimedia, Member: 2002-2005  
Internet Engineering Task Force (IETF), various working groups: 1995-present

### **2. Meeting Support Work**

Internet Engineering Task Force MBone broadcasts: 1995-2005  
Conference MBone broadcasts: Sigcomm '99, and '00  
Interop+Networkworld Network Operations Center (NOC) Team Member: 1995-1997  
ACM Multimedia technical staff: 1994

## **C. University of California Committees**

### **1. Department of Computer Science Committees**

Public Relations: 2005-2006 (chair 2005-2006), 2009-2011 (chair 2009-2011)  
Strategic Planning: 2000-2002, 2003-2006, 2009-2011  
Undergraduate Advising and Affairs: 2006-2007, 2014-2015  
Vice Chair: 2000-2005  
Graduate Admissions: 2000-2005 (chair 2000-2005), 2011-2012  
Graduate Affairs: 2000-2005 (co-chair 2000-2005)  
Teaching Administration: 2000-2005  
Facilities: 1997-2001 (chair 1999-2000), 2006-2007  
External Relations: 1999-2002  
Computer Engineering Public Relations: 2011-2012  
Computer Engineering Awards: 2011-2012  
Computer Engineering Administration/Recruiting: 1998-2001  
Computer Engineering Lab and Computer Support: 1998-2001  
Faculty Recruiting: 1999-2002

Graduate Advising: 1998-1999, 2000-2005

## **2. University Committees**

Member, Campus Budget and Planning: 2013-2015  
Faculty, Cognitive Science Program: 2006-present  
Faculty, Technology Management Program (TMP): 2003-2014  
Faculty, Media Arts and Technology (MAT) Program: 1998-2014  
Faculty, Computer Engineering Degree Program: 1998-present  
Steering Committee, Center for Information Technology and Society (CITS): 2012-present  
Associate Director, Center for Information Technology and Society (CITS): 1999-2012  
Member, Campus Committee on Committees: 2010-2013  
Member, Campus Income and Recharge Committee: 2010-2013  
Member, College of Engineering Executive Committee: 2010-2012 (chair 2011-2012), 2014-2015 (chair 2014-2015)  
Member, Distinguished Teaching Award Committee: 2009, 2010, 2011  
Member, Campus Classroom Design and Renovation Committee: 2003-2010  
Member, ISBER Advisory Committee: 2008-2011  
Member, Fulbright Campus Review Committee: 2007  
Member, Faculty Outreach Grant Program Review Committee: 2007  
Executive Vice Chancellor's Information Technology Fee Committee: 2005-2006  
Council on Research and Instructional Resources: 2003-2006  
Executive Vice Chancellor's Working Group on Graduate Diversity: 2004-2005  
Member, Engineering Pavillion Planning Committee: 2003-2005  
Information Technology Board: 2001-2004  
Executive Committee, Center for Entrepreneurship & Engineering Management (CEEM): 2001-2004

## **3. System Wide Committees**

UCSB Representative to the Committee on Information Technology and Telecommunications Policy (ITTP): 2003-2005  
UCSB Representative to the Executive Committee, Digital Media Innovation (DiMI): 1998-2003

## **D. Georgia Tech Committees and Service (while a graduate student)**

Graduate Student Body President: 1994-1995  
Georgia Tech Executive Board: 1994-1995  
Georgia Tech Alumni Association Executive Committee: 1994-1995  
Dean of Students National Search Committee: 1995  
Institute Strategic Planning Committee: 1994-1996

Cases in last 4 years I have been deposed or testified:

- A deposition in Inter Partes Review of U.S. Patent Nos. 6,286,045 (IPR2015-00657 and IPR2015-00660) and 6,014,698 (IPR2015-00662 and IPR2015-00666) [Google, Inc. v. At Home Bondholders Liquidated Trust]. Finished 12/15.
- A deposition in Inter Partes Review of U.S. Patent Nos. 6,199,076 (IPR2015-00845) and 7,509,178 (IPR2015-00846) [Google, Inc. v. Personal Audio, LLC]. Finished 03/16.
- A deposition in Certain Activity Tracking Devices, Systems, and Components Thereof (US ITC Inv. No. 337-TA-963) [Jawbone v. Fitbit]. Finished 04/16.
- A deposition in Certain Wearable Activity Tracking Devices, Systems, and Components Thereof (US ITC Inv. No. 337-TA-973) [Fitbit v. Jawbone]. Finished 07/16.
- Depositions in Videoshare, LLC V. Google, Inc. and YouTube, LLC (13-cv-990 (GMS), D. Del.). Finished 07/16.
- Depositions and trial testimony in Cisco Systems, Inc. v. Arista Networks, Inc. (5:14-cv-5344-BLF, N.D. Cal.). Finished 12/16.
- A deposition and trial testimony in Sprint Communications Company LP v. Time Warner Cable, Inc. (11-2686-JWL, D. Kan.). Finished 02/17.
- Depositions in Inter Partes Review of U.S. Patent Nos. 6,377,577; 7,023,853; 7,162,537; and 7,224,668 (IPR2016-00303, IPR2016-00306, IPR2016-00308, and IPR2016-0309, respectively) [Arista Networks, Inc. v. Cisco Systems, Inc.]. Finished 01/17.
- A deposition in Intellectual Ventures v. AT&T, CenturyLink, and Windstream (1:13-cv-00116-LY, 1:13-cv-00118-LY, 1:13-cv-00119-LY; W.D. Tex.). Finished 03/17.
- Depositions in Inter Partes Review of U.S. Patent Nos. 7,715,324 and 8,122,102 (IPR2016-01011 and IPR2016-01631, respectively) [Limelight Networks, Inc. v. Akamai Technologies, Inc.]. Finished 03/17.
- Depositions and trial testimony in Certain Network Devices, Related Software and Components Thereof (US ITC Inv. Nos. 337-TA-944 and 337-TA-944E) [Cisco Systems, Inc. v. Arista Networks, Inc.]. Finished 04/17.
- A declaration and Markman testimony in Affinity Labs of Texas, LLC v. Netflix, Inc. (1:15-cv-00849-RP, W.D. Tex.). Finished 04/17.
- A deposition in Inter Partes Review of U.S. Patent No. 7,516,177 (IPR2016-01434) [Oracle America, Inc. and Oracle Corporation and HCC Insurance Holdings, Inc. v. Intellectual Ventures II]. Finished 07/17.
- A deposition in Comcast Cable Communications, LLC v. OpenTV, Inc. and Nagravision SA (3:16-CV-6180-WHA, N.D. Cal.). Finished 07/17.
- Depositions in Thomas C. Sisoian v. International Business Machines Corporation (A-14-CA-565-SS, W.D. Tex.). Finished 07/17.
- A deposition in Inter Partes Review of U.S. Patent Nos. 6,895,449 and 6,470,399 (IPR2017-00713 and IPR2017-00714, respectively) [ZTE (USA) Inc. v. Papst Licensing GmbH & Co. KG.]. Finished 08/17.
- Depositions and trial testimony in Packet Intelligence, LLC v. NetScout Systems, Inc. and Sandvine Corporation (2:16-CV-00230, 2:16-CV-00147, E.D. Tex.). Finished 11/17.



- A deposition in in Certain Two-Way Radio Equipment and Systems, Related Software and Components Thereof (US ITC Inv. Nos. 337-TA-1053) [Motorola v. Hytera]. Finished 11/17.
- A deposition in Sound View Innovations, LLC v. Facebook, Inc. (16-116-RGA, D. Del). Finished 11/17.
- A deposition in Sprint Communications Company LP v. Cox Communications, Inc. (12-487-SLR, 11-2686-JWL, D. Del.). Finished 12/17.
- Depositions in Alacritech, Inc. v. Centurylink Communications LLC; Winstron Corporation, Dell, Inc. (2:16-cv-693-RWS, 2:16-cv-692-RWS, 2:16-cv-695-RWS, E.D. Tex). Finished 12/17.
- A deposition and trial testimony in Sonos, Inc. v. D&M Holding, et al. (14-1330-RGA, D. Del.). Finished 12/17.
- Depositions in Inter Partes Review of U.S. Patent Nos. 7,899,492; 8,050,711; 9,355,611; 8,712,471; 9,286,853; 8,903,451; 8,948,814 and 9,118,794 (IPR2017-00870 through IPR2017-00879) [HTC America, Inc. v. Virginia Innovation Sciences, Inc.]. Finished 01/18.
- Depositions and trial testimony in Certain Network Devices, Related Software and Components Thereof (II) (US ITC Inv. Nos. 337-TA-945 and 337-TA-945M) [Cisco Systems, Inc. v. Arista Networks, Inc.]. Finished 01/18.
- Depositions in Inter Partes Review of U.S. Patent Nos. 8,000,314 (IPR2015-01901); 8,013,732 (IPR2015-01973); 8,754,780 (IPR2016-00984); 8,908,842 (CBM2016-00095); 6,249,516 (IPR2016-01602); 7,697,492 (IPR2016-01895); 7,468,661 (IPR2017-00001); 8,233,471 (IPR2017-00007 and IPR2017-00008); 8,625,496 (IPR2017-00213); 8,013,732 (IPR2017-00216); 6,437,692 (IPR2017-00359); and 8,000,314 (IPR2017-00252) [Emerson Electric Co. v. (S)IPCO, LLC]. Finished 01/18.
- A deposition in D&M Holding, Inc., et al. v. Sonos, Inc. v. (16-141-RGA, D. Del.). Finished 03/18.
- A deposition in Limelight Networks, Inc. v. XO Communication, LLC, Akamai Technologies, Inc., and MIT (3:15cv720-JAG, E.D. Va.). Finished 03/18.
- A depositions in Arista Networks, Inc. v. Cisco Systems, Inc. (5:16-cv-00923-BLF, N.D. Cal.). Finished 08/18.
- Depositions in Inter Partes Review of U.S. Patent No. 9,083,997 (IPR2017-00829 and IPR2017-00830) [Twitter, Inc. v. YouToo Technologies, LLC]. Finished 09/18.
- A deposition in Richard A. Williamson, on behalf of and as a trustee for At Home Bondholders' Liquidating Trust v. Google, LLC (5:15-CV-00966-BLF, N.D. Cal.). Finished 10/18.
- A deposition in Implicit, LLC. v. Palo Alto Networks, Inc. (6:17-cv-182-JRG, E.D. Tex). Finished 10/18.
- A deposition in Inter Partes Review of U.S. Patent Nos. 5,822,523 (IPR2018-00129 and IPR2018-00130) and 6,226,686 (IPR2018-00131 and IPR2018-00132) [Riot Games, Inc. v. Paltalk Holdings, Inc.]. Finished 11/18.
- Depositions in Inter Partes Review of U.S. Patent Nos. 8,116,284 (IPR2018-00128) and 6,591,111 (IPR2018-00176) [Hytera Communications Corp. LTD v. Motorola Solutions, Inc.]. Finished 01/19.
- Depositions in Inter Partes Review of U.S. Patent Nos. 7,237,036 (IPR2017-01391); 7,337,241 (IPR2017-01392); 9,055,104 (IPR2017-01393); 7,124,205 (IPR2017-01405 and IPR2018-00226); 7,673,072 (IPR2017-01406); 8,131,880 (IPR2017-1409 and IPR2017-1410); 8,805,948 (IPR2018-00234); and 7,945,699 (IPR2018-00401) [Intel Corp v. Alacritech, Inc.]. Finished 01/19.

- A claim construction tutorial in Ipsilium, LLC v. Cisco Systems, Inc. (4:17-cv-07179-HSG, N.D. Cal). Finished 03/19.
- Depositions in Inter Partes Review of U.S. Patent Nos. 8,886,739 (IPR2018-00200); 9,306,885 (IPR2018-00312); 9,315,155 (IPR2018-00369); 9,306,886 (IPR2018-00397); 8,935,351 (IPR2018-00404); 9,338,111 (IPR2018-00408); 9,413,711 (IPR2018-00416 and IPR2018-00439); 9,313,157 (IPRs018-00455); and 9,313,156 (IPR2018-00458) [Snap, Inc. v. Vaporstream, Inc.]. Finished 04/19.
- Depositions and trial testimony in TQ Delta, LLC v. 2Wire, Inc. (1:13-cv-01835-RGA, D. Del.). 05/19.
- Depositions in Rmail Limited, et al. v. Amazon.com, Inc.; Docusign, Inc.; Right Signature, LLC, et al.; and Adobe Systems Inc., et al. (2:10-cv-258-JRG, 2:11-cv-299-JRG, 2:11-cv-300-JRG, 2:11-cv-325-JRG, E.D. Tex.). Finished 06/19.
- Depositions and trial testimony in Certain Wireless Mesh Networking Products and Related Components Thereof (US ITC Inv. No. 337-TA-1131) [SIPCO, LLC v. Emerson Electric Co]. Finished 09/19.
- A deposition in Packet Intelligence v. Nokia of America Corporation (2:18-cv-382-JRG, E.D. Tex.). Finished 09/19.
- Depositions in TQ Delta, LLC v. AdTran, Inc. (14-cv-954-RGA and 15-cv-121-RGA, D. Del). Finished 10/19.
- 
- A deposition in Twilio, Inc. v. Telesign Corporation (5:16-CV-06925-LHK-SVK, N.D. Cal)
- Depositions and claim construction tutorial in Netfuel, Inc. v. Cisco Systems, Inc. (5:18-cv-2352-EJD, N.D. Cal)
- A deposition and Markman tutorial in Blackberry Limited v. Facebook, Inc. and Snap, Inc. (2:18-cv-01844 GW(KSx) and 2:18-cv-02693-GW(KSx), C.D. Cal.)
- A deposition in Sony Music Entertainment, et al. v. Cox Communicaions, Inc. et al. (1:18-cv-00950-LO-JFA, E.D. Va.)
- A deposition in Finjan, Inc. v. Cisco Systems, Inc. (5:17-cv-00072-BLF-SVK, N.D. Cal)
- A deposition in Implicit, LLC v. NetScout Systems, Inc. and Sandvine Corporation (2:18-cv-0053-JRG and 2::18-cv-0054-JRG, E.D. Tex).
- A deposition in Certain Digital Video Receivers, Broadband Gateways, and Related Hardware and Software Components (US ITC Inv. No. 337-TA-1158) [Rovi Guides, Inc. v. Comcast Cable Communications, LLC].



# Appendix 2

Materials Considered List  
Invalidity Expert Report of Dr. Kevin C. Almeroth

**Case Documents**

- NetFuel's First Amended Complaint, Dkt. 60, January 31, 2019
- Cisco's Answer to First Amended Complaint, Dkt. 61, February 14, 2019
- NetFuel's Opening Claim Construction Brief and exhibits, Dkt. 50, January 8, 2019
- Cisco's Responsive Claim Construction Brief and exhibits, Dkt. 56, January 22, 2019
- NetFuel's Reply Claim Construction Brief and exhibits, Dkt. 59, January 29, 2019
- Second Amended Joint Claim Construction and Prehearing Statement, Dkt. 68, March 1, 2019
- Dkt No. 71, Markman Order, dated April 11, 2019
- NetFuel's Disclosure of Asserted Claims and Infringement Contentions Pursuant Local Rule 3-1, and exhibits attached thereto, served August 2, 2018
- NetFuel's Amended Infringement Contentions, and exhibits attached thereto, served July 1, 2019
- Opening Expert Report of Dr. Kevin Almeroth regarding Non-Infringement and materials cited within, May 9, 2019
- Opening Expert Report of Dr. Aviel Rubin regarding Non-Infringement and materials cited within, May 9, 2019
- Rebuttal Expert Report of Dr. Kevin Almeroth regarding Non-Infringement and materials cited within, June 6, 2019
- Rebuttal Expert Report of Dr. Aviel Rubin regarding Non-Infringement and materials cited within, June 6, 2019
- Preliminary Invalidity Contentions and Statement Regarding Document Disclosures Pursuant to Patent Local Rules 3-3 and 3-4, exhibits attached thereto, and documents cited within, served September 18, 2018
- Cisco's Supplemental Invalidity Charts, and documents cited within, served August, 21, 2019
- Documents cited within this expert report and within exhibits attached thereto

**Patents and File History**

- United States Patent No. 7,747,730
- United States Patent No. 9,663,659

Materials Considered List  
Invalidity Expert Report of Dr. Kevin C. Almeroth

- File History of United States Patent No. 7,747,730
- File History of United States Patent No. 9,663,659

**Public Documents**

- Cisco Systems, Inc., NetRanger Intrusion Detection System Technical Overview (1998), *available at* [http://storage.library.opu.ua/online/external/cisco/products/778/security/netranger/ntran\\_tc.htm](http://storage.library.opu.ua/online/external/cisco/products/778/security/netranger/ntran_tc.htm)
- IETF Network Working Group RFC 2328, “OSPF Version 2,” (April 1998), *available at* <https://tools.ietf.org/html/rfc2328>, OSPF v2

**Bates Numbered Documents**

- |                   |                   |                   |
|-------------------|-------------------|-------------------|
| • CSI-NF-00000001 | • CSI-NF-00000002 | • CSI-NF-00000003 |
| • CSI-NF-00000004 | • CSI-NF-00000005 | • CSI-NF-00000006 |
| • CSI-NF-00000011 | • CSI-NF-00000012 | • CSI-NF-00000013 |
| • CSI-NF-00000014 | • CSI-NF-00000015 | • CSI-NF-00000016 |
| • CSI-NF-00000017 | • CSI-NF-00000018 | • CSI-NF-00000019 |
| • CSI-NF-00000020 | • CSI-NF-00000021 | • CSI-NF-00000022 |
| • CSI-NF-00000023 | • CSI-NF-00000024 | • CSI-NF-00000025 |
| • CSI-NF-00000026 | • CSI-NF-00000027 | • CSI-NF-00000028 |
| • CSI-NF-00000029 | • CSI-NF-00000030 | • CSI-NF-00000031 |
| • CSI-NF-00000036 | • CSI-NF-00000040 | • CSI-NF-00000044 |
| • CSI-NF-00000045 | • CSI-NF-00000046 | • CSI-NF-00000047 |
| • CSI-NF-00000048 | • CSI-NF-00000049 | • CSI-NF-00000050 |
| • CSI-NF-00000051 | • CSI-NF-00000052 | • CSI-NF-00000053 |
| • CSI-NF-00000054 | • CSI-NF-00000055 | • CSI-NF-00000056 |
| • CSI-NF-00000057 | • CSI-NF-00000058 | • CSI-NF-00000059 |
| • CSI-NF-00000060 | • CSI-NF-00000061 | • CSI-NF-00000062 |
| • CSI-NF-00000063 | • CSI-NF-00000064 | • CSI-NF-00000065 |
| • CSI-NF-00000066 | • CSI-NF-00000067 | • CSI-NF-00000068 |

Materials Considered List  
Invalidity Expert Report of Dr. Kevin C. Almeroth

- |                   |                   |                   |
|-------------------|-------------------|-------------------|
| • CSI-NF-00000069 | • CSI-NF-00000070 | • CSI-NF-00000071 |
| • CSI-NF-00000072 | • CSI-NF-00000073 | • CSI-NF-00000074 |
| • CSI-NF-00000075 | • CSI-NF-00000076 | • CSI-NF-00000077 |
| • CSI-NF-00000078 | • CSI-NF-00000079 | • CSI-NF-00000080 |
| • CSI-NF-00000081 | • CSI-NF-00000082 | • CSI-NF-00000083 |
| • CSI-NF-00000084 | • CSI-NF-00000085 | • CSI-NF-00000086 |
| • CSI-NF-00000087 | • CSI-NF-00000088 | • CSI-NF-00000089 |
| • CSI-NF-00000090 | • CSI-NF-00000091 | • CSI-NF-00000092 |
| • CSI-NF-00000093 | • CSI-NF-00000094 | • CSI-NF-00000095 |
| • CSI-NF-00000096 | • CSI-NF-00000097 | • CSI-NF-00000098 |
| • CSI-NF-00000099 | • CSI-NF-00000100 | • CSI-NF-00000819 |
| • CSI-NF-00000820 | • CSI-NF-00000821 | • CSI-NF-00000822 |
| • CSI-NF-00000823 | • CSI-NF-00000824 | • CSI-NF-00000825 |
| • CSI-NF-00000826 | • CSI-NF-00000827 | • CSI-NF-00000828 |
| • CSI-NF-00000829 | • CSI-NF-00000830 | • CSI-NF-00000831 |
| • CSI-NF-00000832 | • CSI-NF-00000833 | • CSI-NF-00000834 |
| • CSI-NF-00000835 | • CSI-NF-00000836 | • CSI-NF-00000837 |
| • CSI-NF-00000838 | • CSI-NF-00000839 | • CSI-NF-00000840 |
| • CSI-NF-00000841 | • CSI-NF-00000842 | • CSI-NF-00000843 |
| • CSI-NF-00000844 | • CSI-NF-00000845 | • CSI-NF-00000846 |
| • CSI-NF-00000847 | • CSI-NF-00000848 | • CSI-NF-00000849 |
| • CSI-NF-00000850 | • CSI-NF-00000851 | • CSI-NF-00000852 |
| • CSI-NF-00000853 | • CSI-NF-00000854 | • CSI-NF-00000855 |
| • CSI-NF-00000856 | • CSI-NF-00000857 | • CSI-NF-00000858 |
| • CSI-NF-00000859 | • CSI-NF-00000860 | • CSI-NF-00000861 |
| • CSI-NF-00000862 | • CSI-NF-00000863 | • CSI-NF-00000864 |
| • CSI-NF-00000865 | • CSI-NF-00000866 | • CSI-NF-00000867 |
| • CSI-NF-00000868 | • CSI-NF-00000869 | • CSI-NF-00000870 |
| • CSI-NF-00000871 | • CSI-NF-00000872 | • CSI-NF-00000873 |

Materials Considered List  
Invalidity Expert Report of Dr. Kevin C. Almeroth

- |                   |                   |                   |
|-------------------|-------------------|-------------------|
| • CSI-NF-00000874 | • CSI-NF-00000875 | • CSI-NF-00000876 |
| • CSI-NF-00000877 | • CSI-NF-00000878 | • CSI-NF-00000879 |
| • CSI-NF-00000880 | • CSI-NF-00000881 | • CSI-NF-00000882 |
| • CSI-NF-00000883 | • CSI-NF-00000884 | • CSI-NF-00000885 |
| • CSI-NF-00000886 | • CSI-NF-00000887 | • CSI-NF-00000888 |
| • CSI-NF-00000889 | • CSI-NF-00000890 | • CSI-NF-00000891 |
| • CSI-NF-00000892 | • CSI-NF-00000893 | • CSI-NF-00000894 |
| • CSI-NF-00000895 | • CSI-NF-00000896 | • CSI-NF-00000897 |
| • CSI-NF-00000898 | • CSI-NF-00000899 | • CSI-NF-00000900 |
| • CSI-NF-00000901 | • CSI-NF-00000902 | • CSI-NF-00000903 |
| • CSI-NF-00000904 | • CSI-NF-00000905 | • CSI-NF-00000906 |
| • CSI-NF-00000907 | • CSI-NF-00000908 | • CSI-NF-00000909 |
| • CSI-NF-00000910 | • CSI-NF-00000911 | • CSI-NF-00000912 |
| • CSI-NF-00000913 | • CSI-NF-00000914 | • CSI-NF-00000915 |
| • CSI-NF-00000916 | • CSI-NF-00000917 | • CSI-NF-00000918 |
| • CSI-NF-00000919 | • CSI-NF-00000920 | • CSI-NF-00000921 |
| • CSI-NF-00000922 | • CSI-NF-00000923 | • CSI-NF-00000924 |
| • CSI-NF-00000925 | • CSI-NF-00000926 | • CSI-NF-00000927 |
| • CSI-NF-00000928 | • CSI-NF-00000929 | • CSI-NF-00000930 |
| • CSI-NF-00000931 | • CSI-NF-00000932 | • CSI-NF-00000933 |
| • CSI-NF-00000934 | • CSI-NF-00000935 | • CSI-NF-00000936 |
| • CSI-NF-00000937 | • CSI-NF-00000938 | • CSI-NF-00000939 |
| • CSI-NF-00000940 | • CSI-NF-00000941 | • CSI-NF-00000942 |
| • CSI-NF-00000943 | • CSI-NF-00000944 | • CSI-NF-00000945 |
| • CSI-NF-00000946 | • CSI-NF-00000947 | • CSI-NF-00000948 |
| • CSI-NF-00000949 | • CSI-NF-00000950 | • CSI-NF-00000951 |
| • CSI-NF-00000952 | • CSI-NF-00000953 | • CSI-NF-00000954 |
| • CSI-NF-00000955 | • CSI-NF-00001121 | • CSI-NF-00001122 |
| • CSI-NF-00001123 | • CSI-NF-00001124 | • CSI-NF-00001125 |

Materials Considered List  
Invalidity Expert Report of Dr. Kevin C. Almeroth

- |                   |                   |                   |
|-------------------|-------------------|-------------------|
| • CSI-NF-00001126 | • CSI-NF-00001127 | • CSI-NF-00001128 |
| • CSI-NF-00001129 | • CSI-NF-00001130 | • CSI-NF-00001131 |
| • CSI-NF-00001132 | • CSI-NF-00001133 | • CSI-NF-00001134 |
| • CSI-NF-00001135 | • CSI-NF-00001136 | • CSI-NF-00001137 |
| • CSI-NF-00001138 | • CSI-NF-00001139 | • CSI-NF-00001140 |
| • CSI-NF-00001141 | • CSI-NF-00001142 | • CSI-NF-00001143 |
| • CSI-NF-00001144 | • CSI-NF-00001145 | • CSI-NF-00001146 |
| • CSI-NF-00001147 | • CSI-NF-00001148 | • CSI-NF-00001149 |
| • CSI-NF-00001150 | • CSI-NF-00001151 | • CSI-NF-00001152 |
| • CSI-NF-00001153 | • CSI-NF-00001154 | • CSI-NF-00001155 |
| • CSI-NF-00001156 | • CSI-NF-00001157 | • CSI-NF-00001158 |
| • CSI-NF-00001159 | • CSI-NF-00001160 | • CSI-NF-00001161 |
| • CSI-NF-00001162 | • CSI-NF-00001163 | • CSI-NF-00001164 |
| • CSI-NF-00001165 | • CSI-NF-00001166 | • CSI-NF-00001167 |
| • CSI-NF-00001168 | • CSI-NF-00001169 | • CSI-NF-00001170 |
| • CSI-NF-00001171 | • CSI-NF-00001172 | • CSI-NF-00001173 |
| • CSI-NF-00001174 | • CSI-NF-00001175 | • CSI-NF-00001176 |
| • CSI-NF-00001177 | • CSI-NF-00001178 | • CSI-NF-00001179 |
| • CSI-NF-00001180 | • CSI-NF-00001181 | • CSI-NF-00001182 |
| • CSI-NF-00001183 | • CSI-NF-00001184 | • CSI-NF-00001185 |
| • CSI-NF-00001186 | • CSI-NF-00001187 | • CSI-NF-00001188 |
| • CSI-NF-00001189 | • CSI-NF-00001190 | • CSI-NF-00001191 |
| • CSI-NF-00001192 | • CSI-NF-00001193 | • CSI-NF-00001194 |
| • CSI-NF-00001195 | • CSI-NF-00001196 | • CSI-NF-00001197 |
| • CSI-NF-00001198 | • CSI-NF-00001199 | • CSI-NF-00001200 |
| • CSI-NF-00001201 | • CSI-NF-00001202 | • CSI-NF-00001203 |
| • CSI-NF-00001204 | • CSI-NF-00001205 | • CSI-NF-00001206 |
| • CSI-NF-00001207 | • CSI-NF-00001208 | • CSI-NF-00001209 |
| • CSI-NF-00001210 | • CSI-NF-00001211 | • CSI-NF-00001212 |

Materials Considered List  
Invalidity Expert Report of Dr. Kevin C. Almeroth

- |                   |                   |                   |
|-------------------|-------------------|-------------------|
| • CSI-NF-00001213 | • CSI-NF-00001214 | • CSI-NF-00001215 |
| • CSI-NF-00001216 | • CSI-NF-00001217 | • CSI-NF-00001218 |
| • CSI-NF-00001219 | • CSI-NF-00001220 | • CSI-NF-00001221 |
| • CSI-NF-00001222 | • CSI-NF-00001223 | • CSI-NF-00001224 |
| • CSI-NF-00001225 | • CSI-NF-00001226 | • CSI-NF-00001227 |
| • CSI-NF-00001228 | • CSI-NF-00001229 | • CSI-NF-00001230 |
| • CSI-NF-00001231 | • CSI-NF-00001232 | • CSI-NF-00001233 |
| • CSI-NF-00001234 | • CSI-NF-00001235 | • CSI-NF-00001236 |
| • CSI-NF-00001237 | • CSI-NF-00001238 | • CSI-NF-00001239 |
| • CSI-NF-00001240 | • CSI-NF-00001241 | CSI-NF-00001242   |
| • CSI-NF-00001243 | • CSI-NF-00001244 | CSI-NF-00001245   |
| • CSI-NF-00001246 | • CSI-NF-00001247 | CSI-NF-00001248   |
| • CSI-NF-00001249 | • CSI-NF-00001250 | • CSI-NF-00001251 |
| • CSI-NF-00001252 | • CSI-NF-00001253 | • CSI-NF-00001254 |
| • CSI-NF-00001255 | • CSI-NF-00001256 | • CSI-NF-00001257 |
| • CSI-NF-00001258 | • CSI-NF-00001259 | • CSI-NF-00001260 |
| • CSI-NF-00001261 | • CSI-NF-00001262 | • CSI-NF-00001263 |
| • CSI-NF-00001264 | • CSI-NF-00001265 | • CSI-NF-00001266 |
| • CSI-NF-00001267 | • CSI-NF-00001268 | • CSI-NF-00001269 |
| • CSI-NF-00001270 | • CSI-NF-00001271 | • CSI-NF-00001272 |
| • CSI-NF-00001273 | • CSI-NF-00001274 | • CSI-NF-00001275 |
| • CSI-NF-00001276 | • CSI-NF-00001277 | • CSI-NF-00001278 |
| • CSI-NF-00001279 | • CSI-NF-00001280 | • CSI-NF-00001281 |
| • CSI-NF-00001282 | • CSI-NF-00001283 | • CSI-NF-00001284 |
| • CSI-NF-00001285 | • CSI-NF-00001286 | • CSI-NF-00001287 |
| • CSI-NF-00001288 | • CSI-NF-00001289 | • CSI-NF-00001290 |
| • CSI-NF-00001291 | • CSI-NF-00001292 | • CSI-NF-00001293 |
| • CSI-NF-00001294 | • CSI-NF-00001295 | • CSI-NF-00001296 |
| • CSI-NF-00001297 | • CSI-NF-00001298 | • CSI-NF-00001299 |

Materials Considered List  
Invalidity Expert Report of Dr. Kevin C. Almeroth

- |                   |                   |                   |
|-------------------|-------------------|-------------------|
| • CSI-NF-00001300 | • CSI-NF-00001301 | • CSI-NF-00001302 |
| • CSI-NF-00001303 | • CSI-NF-00001304 | • CSI-NF-00001305 |
| • CSI-NF-00001306 | • CSI-NF-00001307 | • CSI-NF-00001308 |
| • CSI-NF-00001309 | • CSI-NF-00001310 | • CSI-NF-00001311 |
| • CSI-NF-00001312 | • CSI-NF-00001313 | • CSI-NF-00001314 |
| • CSI-NF-00001315 | • CSI-NF-00001316 | • CSI-NF-00001317 |
| • CSI-NF-00001318 | • CSI-NF-00001319 | • CSI-NF-00001320 |
| • CSI-NF-00001321 | • CSI-NF-00001322 | • CSI-NF-00001323 |
| • CSI-NF-00001324 | • CSI-NF-00001325 | • CSI-NF-00001326 |
| • CSI-NF-00001327 | • CSI-NF-00001328 | • CSI-NF-00001329 |
| • CSI-NF-00001330 | • CSI-NF-00001331 | • CSI-NF-00001332 |
| • CSI-NF-00001333 | • CSI-NF-00001334 | • CSI-NF-00001335 |
| • CSI-NF-00001336 | • CSI-NF-00001337 | • CSI-NF-00001338 |
| • CSI-NF-00001339 | • CSI-NF-00001340 | • CSI-NF-00001341 |
| • CSI-NF-00001342 | • CSI-NF-00011567 | • CSI-NF-00100307 |
| • CSI-NF-00100308 | • CSI-NF-00100309 | • CSI-NF-00100310 |
| • CSI-NF-00100311 | • CSI-NF-00100312 | • CSI-NF-00100313 |
| • CSI-NF-00100314 | • CSI-NF-00100315 | • CSI-NF-00100316 |
| • CSI-NF-00100317 | • CSI-NF-00100318 |                   |